Advanced Card Systems Ltd.
Card & Reader Technologies

ACR1552U Series

Reference Manual V1.05

Subject to change without prior notice

info@acs.com.hk
www.acs.com.hk

Table of Contents

1.0. Introduction 4

2.0. Features 5

3.0. ACR1552U Architecture 6

3.1. Reader Block Diagram 6

3.2. Communication between PC/SC driver and PICC and SAM 7

4.0. Hardware Design 8

4.1. USB 8

4.1.1. Communication Parameters 8
4.1.2. Endpoints 8

4.2. Contactless Smart Card Interface 8

4.2.1. Carrier Frequency 8
4.2.2. Card Polling 8

4.3. User Interface 9

4.3.1. Buzzer and LED 9

5.0. Software Design 10

5.1. PCSC API 10

5.1.1. SCardEstablishContext 10
5.1.2. SCardListReaders 11
5.1.3. SCardConnect 12
5.1.4. SCardControl 13
5.1.5. SCardTransmit 15
5.1.6. SCardDisconnect 17
5.1.7. APDU Flow 18
5.1.8. Escape Command Flow 19

5.2. Contact Smart Card Protocol 20

5.2.1. ACOS6-SAM Commands 20

5.3. Contactless Smart Card Protocol 31

5.3.1. ATR Generation 31
5.3.2. APDU, Pseudo APDU and Card Native Command 34
5.3.3. PCSC Pseudo APDU (with Proprietary Extension) for PICC 34
5.3.4. Proprietary Pseudo APDU for PICC 55
5.3.5. Accessing PCSC-Compliant tags (ISO14443-4) 58
5.3.6. Accessing FeliCa tags 60
5.3.8. Accessing ISO15693 tags 61
5.3.9. Supported PICC ATR 67

6.0. Escape Command 70

6.1. Escape Command for PICC 70

6.1.1. RF Control [E0 00 00 25 01 …] 70
6.1.2. Get PCD/PICC Status [E0 00 00 25 00] 71
6.1.3. Get Polling/ATR Option [E0 00 00 23 00] 72
6.1.4. Set Polling/ATR Option [E0 00 00 23 01 …] 72
6.1.5. Get PICC Polling Type [E0 00 01 20 00] 73
6.1.6. Set PICC Polling Type [E0 00 01 20 02 …] 73
6.1.7. Get Auto PPS [E0 00 00 24 00] 74
6.1.8. Set Auto PPS [E0 00 00 24 01 …] 74
6.1.9. Read PICC Type [E0 00 00 35 00] 75
6.1.10. Get RF Power Setting [E0 00 00 50 00] 77
6.1.11. Set RF Power Setting [E0 00 00 50 01 …] 77

6.1.12. Escape Command for PICC – HID Keyboard 78
6.1.13. Escape Command for PICC – Card Emulation 83
6.1.14. Escape Command for PICC – Discovery Mode 89

6.2. Escape Command for Peripheral Control and Other 90
6.2.1. Get Firmware Version [E0 00 00 18 …] 90
6.2.2. Get Serial Number [E0 00 00 33 00] 90
6.2.3. Set S/N in USB Descriptor [E0 00 00 F0] 91
6.2.4. Set Buzzer Control - Single Time [E0 00 00 28 01 …] 91
6.2.5. Set Buzzer Control - Repeatable [E0 00 00 28 03 …] 92
6.2.6. Get LED Status [E0 00 00 29 00] 92
6.2.7. Set LED Control [E0 00 00 29 01 …] 93
6.2.8. Get UI Behaviour [E0 00 00 21 00] 93
6.2.9. Set UI Behaviour [E0 00 00 21 01 …] 93

Appendix A. NDEF Message 95

List of Figures

Figure 1 : ACR1552U Reader Block Diagram 6
Figure 2 : ACR1552U Architecture 7
Figure 3 : ACR1552U APDU Flow 18
Figure 4 : ACR1552U Escape Command Flow 19

List of Tables

Table 1 : USB Interface Wiring 8
Table 2 : Buzzer and LED Indicator 9
Table 3 : MIFARE Classic 1K Memory Map 37
Table 4 : MIFARE Classic 4K Memory Map 38
Table 5 : MIFARE Ultralight Memory Map 39
Table 6 : NFC Forum Type 2 Tag Memory Map (2000 bytes) 83
Table 7 : FeliCa Memory Map (160 bytes) 84

1.0. Introduction

Continuing the success of the ACR1252U Series, ACR1552U Series is the USB NFC Reader developed based on 13.56Mhz contactless technology. It is a contactless reader that can access contactless smart cards following the ISO 14443, ISO 15693 & ISO 18092 standards, MIFARE® (T=CL), FeliCa, NFC Tags, SRI/SRIX, CTS, Innovatron, Picopass and Topaz Card are also supported.

ACR1552U Series is capable of the three modes of NFC, namely: card reader/writer, card emulation and keyboard emulation. It also has a built-in SAM slot for added security in both contact and contactless applications.

Compliant with both CCID and PC/SC, this plug-and-play USB NFC device allows interoperability with different devices and applications. It is thus ideal for unconventional marketing and advertising applications like smart posters.

With additional features such as Keyboard Emulation, ACR1552 Series is a highly cost-effective, powerful all-in-one device that offers convenience and flexibility to many smart card applications.

This document is applicable to ACR1552U Series,

- ACR1552U-M*, USB NFC Reader IV
- ACM1552U-Y*, USB NFC Reader Module with Detachable Antenna Board
- ACM1552U-Z*, Small NFC Reader Module

Notes: Availability of certain features, such as the SAM slot and Buzzer, may vary depending on the product model.

2.0. Features

- USB Full Speed Interface
- CCID-compliant
- Smart Card Reader:
    - Contactless Interface:
        - Read/Write speed of up to 26kbps ISO 15693 & 848 kbps (ISO 14443) card types
        - Built-in antenna for contactless tag access, with card reading distance of up to 70 mm (depending on tag type)
        - Supports ISO 15693 card types
        - Supports ISO 14443 Part 4 Type A and B cards and MIFARE series
        - Built-in anti-collision feature
        - Supports extended APDU (max. 64 KB)
    - SAM Interface:
        - One SAM Slot
        - Supports ISO 7816 Class A SAM cards
- Application Programming Interface:
    - Supports PC/SC
    - Supports CT-API (through wrapper on top of PC/SC)
- Built-in Peripherals:
    - Two user-controllable LEDs (Blue and Green)
    - User-controllable buzzer
- USB Firmware Upgradability
- Supports Android™ 3.1 and later[1]
- Compliant with the following standards:
    - ISO 14443
    - ISO 15693
    - ISO 7816
    - PC/SC
    - CCID
    - CE
    - UKCA
    - FCC
    - RoHS
    - REACH
    - Microsoft® WHQL

[1] *Uses an ACS-defined Android Library*

3.0. ACR1552U Architecture

3.1. Reader Block Diagram

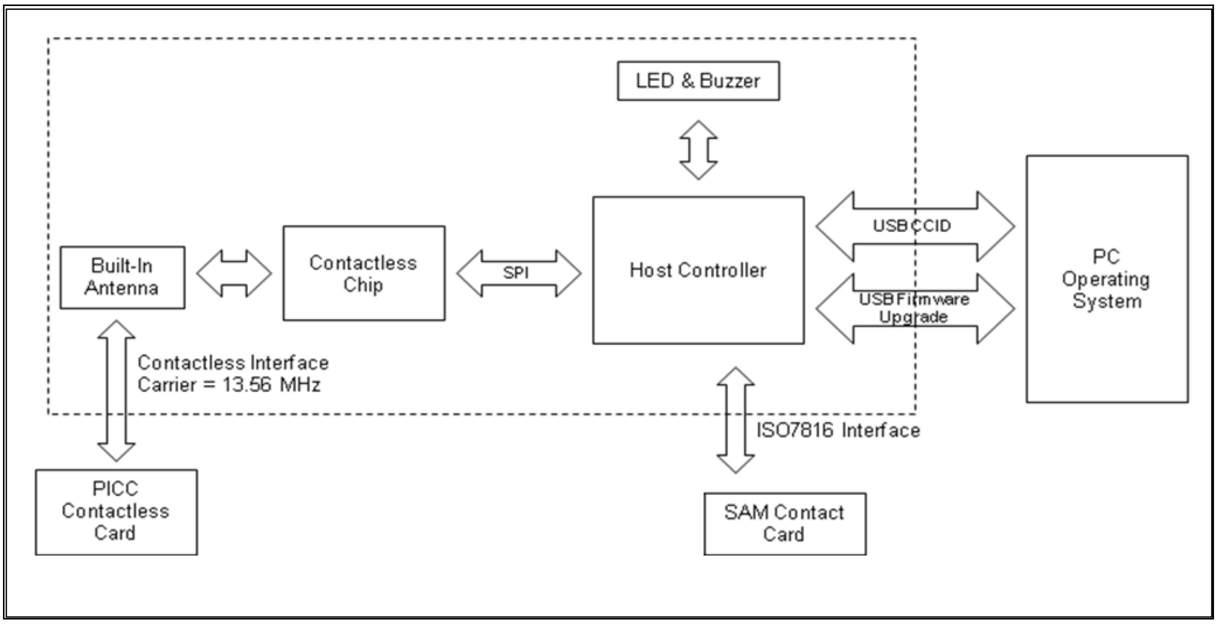


Figure 1: ACR1552U Reader Block Diagram

3.2. Communication between PC/SC driver and PICC and SAM

The protocol being used between the ACR1552U and the PC is CCID. All communications between PICC and SAM are PC/SC-compliant.

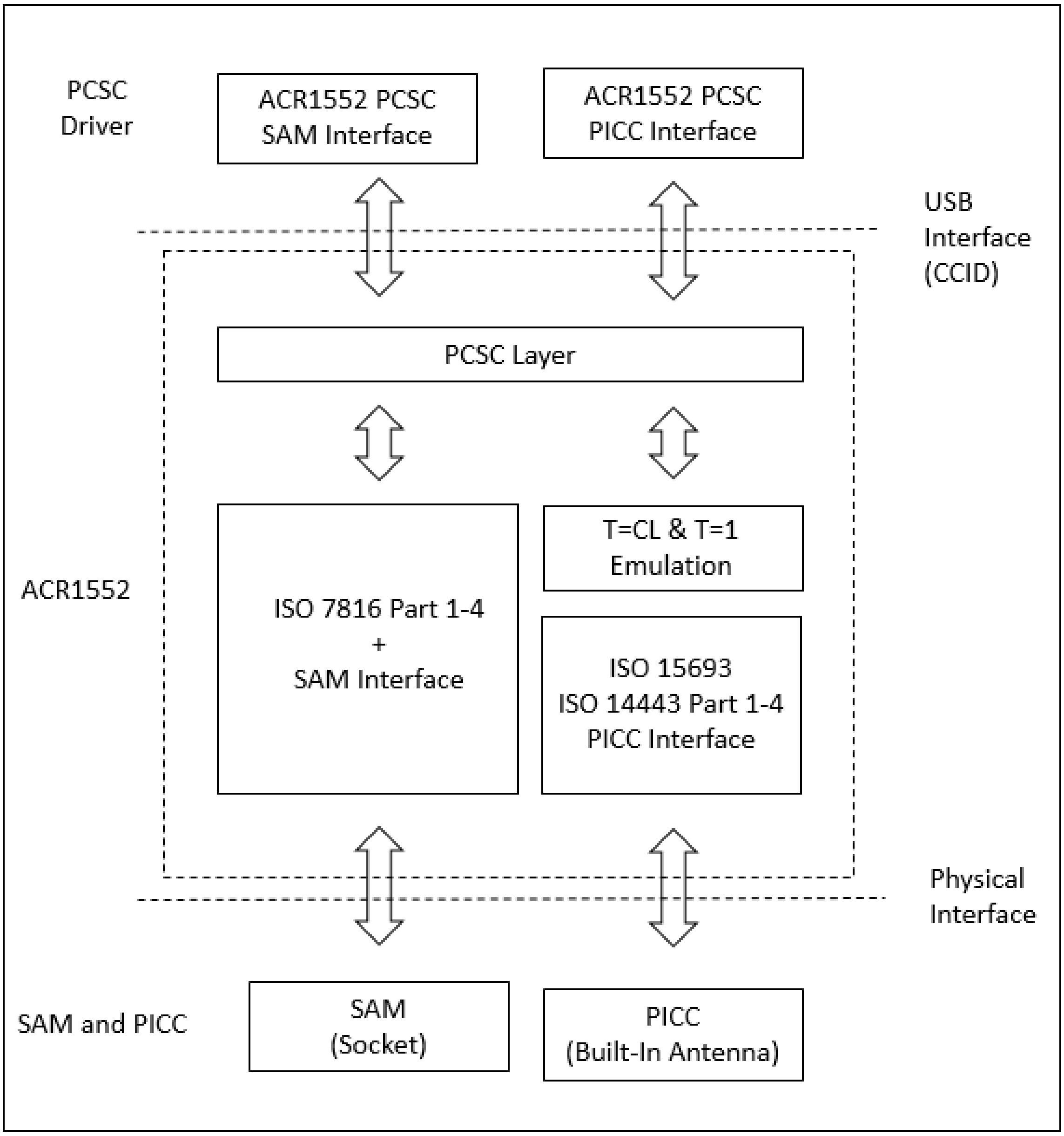


Figure 2: ACR1552U Architecture

4.0. Hardware Design

4.1. USB

The ACR1552U connects to a computer through USB following the USB standard.

4.1.1. Communication Parameters

The ACR1552U connects to a computer through USB as specified in the USB Specification 2.0. The ACR1552U works in full-speed mode, i.e. 12 Mbps.

Pin	Signal	Function
1	V_{BUS}	+5 V power supply for the reader
2	D-	Differential signal transmits data between ACR1552U and PC
3	D+	Differential signal transmits data between ACR1552U and PC
4	GND	Reference voltage level for power supply

Table 1: USB Interface Wiring

Note: *The device driver should be installed for the ACR1552U to function properly through USB interface.*

4.1.2. Endpoints

The ACR1552U uses the following endpoints to communicate with the host computer:

Control Endpoint – For setup and control purposes.

Bulk-OUT – For commands to be sent from the host to the ACR1552U (data packet size is 64 bytes).

Bulk-IN – For response to be sent from the ACR1552U to the host (data packet size is 64 bytes).

Interrupt-IN – For card status message to be sent from the ACR1552U to the host (data packet size is 8 bytes).

4.2. Contactless Smart Card Interface

The interface between the ACR1552U and the contactless card follows the specifications of ISO 14443 with certain restrictions or enhancements to increase the practical functionality of the ACR1552U.

4.2.1. Carrier Frequency

The carrier frequency for the ACR1552U is 13.56 MHz.

4.2.2. Card Polling

The ACR1552U automatically polls the contactless cards that are within the field. ISO 14443-4 Type A, ISO 14443-4 Type B, ISO 15693 and MIFARE cards are supported.

4.3. User Interface

4.3.1. Buzzer and LED

The monotone buzzer and LEDs used for showing the state of the contactless interfaces. The Blue LED is used for showing PICC status.

Reader States	Buzzer	Blue LED (PICC)
1. Plug in the reader	Beep Once	🔵 >> 🟢 >> 🔵
2. Standby (Contactless Polling, no PICC card)	Off	🔵
3. Standby (No Polling)	Off	Off
4. Contactless Card is tapped	Beep Once	🔵
5. Contactless Card is presence	Off	🔵
6. Contactless Card is removed	Off	🔵
7. Contactless Card is communicating	Off	Fast Blinking

Table 2: Buzzer and LED Indicator

5.0. Software Design

5.1. PCSC API

This section will describe some of the PCSC API for application programming usage. For more details, please refer to Microsoft MSDN Library or PCSC workgroup.

5.1.1. SCardEstablishContext

The SCardEstablishContext function establishes the resource manager context within which database operations are performed.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379479%28v=vs.85%29.aspx

This function should be performed first before any other PCSC operation.

Example:

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext;
int retCode;
void main ()
{
   // To establish the resource manager context and assign it to “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // Further PCSC operation can be performed
   }
}
```

5.1.2. SCardListReaders

The SCardListReaders function provides the list of readers within a set of named reader groups, eliminating duplicates.

The caller supplies a list of reader groups, and receives the list of readers within the named groups. Unrecognized group names are ignored. This function only returns readers within the named groups that are currently attached to the system and available for use.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379793%28v=vs.85%29.aspx
Example:

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext; // Resource manager context
int retCode;
char readerName [256]; // List reader name

void main ()
{
   // To establish the resource manager context and assign to “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // List the available reader which can be used in the system
      retCode = SCardListReaders (hContext,
                  NULL,
                  readerName,
                  &size);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Listing reader fail
      }
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      }
   }
}
```

5.1.3. SCardConnect

The SCardConnect function establishes a connection (using a specific resource manager context) between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379473%28v=vs.85%29.aspx

Example:

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT      hContext;         // Resource manager context
SCARDHANDLE       hCard;            // Card context handle
unsigned long     dwActProtocol;    // Establish active protocol
int               retCode;
char              readerName [256]; // List reader name
char              rName [256];      // Reader name for connection

void main ()
{
   …
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      rName = "ACS ACR1552 1S CL Reader PICC 0"; // Depends on what
                                                reader be used
                                                // Should connect to
                                                PICC interface
         retCode = SCardConnect(hContext,
               rName,
               SCARD_SHARE_SHARED,
               SCARD_PROTOCOL_T0,
               &hCard,
               &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (May be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
         }
      }
   }
```

5.1.4. SCardControl

The SCardControl function gives you direct control of the reader. You can call it any time after a successful call to SCardConnect and before a successful call to SCardDisconnect. The effect on the state of the reader depends on the control code.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379474%28v=vs.85%29.aspx

Note: *Commands from* ***Escape Command*** *use this API for sending.*

Example:

```
#define SCARD_SCOPE_USER   0

#define EscapeCommand 0x310000 + 3500*4
   SCARDCONTEXT         hContext;         // Resource manager context
   SCARDHANDLE          hCard;            // Card context handle
   unsigned long        dwActProtocol;    // Established active protocol
   int                  retCode;
   char                 readerName [256]; // Lists reader name
   char                 rName [256];      // Reader name for connection
   BYTE                 SendBuff[262],    // APDU command buffer
                        RecvBuff[262];    // APDU response buffer
   BYTE                 FWVersion [20],   // For storing firmware
                                             version message
   BYTE                 ResponseData[50]; // For storing card response
   DWORD                SendLen,          // APDU command length
                        RecvLen;          // APDU response length
void main ()
   {
      …
      rName = "ACS ACR1552 1S CL Reader PICC 0";   // Depends on what
                                                   reader will be used
                                                   // Should connect to
                                                   PICC interface
         retCode = SCardConnect(hContext,
            rName,
            SCARD_SHARE_DIRECT,
            SCARD_PROTOCOL_T0| SCARD_PROTOCOL_T1,
            &hCard,
            &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (may be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
            RecvLen = 262;
            // Get firmware version
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x18;
            SendBuff[4] = 0x00;
            SendLen = 5;
            retCode = SCardControl ( hCard,
                     EscapeCommand,
                     SendBuff,
                     SendLen,
                     RecvBuff,
                     RecvLen,
                     &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending successful
               // The RecvBuff stores the firmware version message.
               for (int i=0;i< RecvLen-5;i++)
               {
                  FWVersion[i] = RecvBuff [5+i];
               }
            }
            // Connection successful
            RecvLen = 262;

            // Turn Green LED on, turn Red LED off
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x29;
            SendBuff[4] = 0x01;
            SendBuff[5] = 0x02;  // Green LED On, Red LED off
            SendLen = 6;
            retCode = SCardControl ( hCard,
                     EscapeCommand,
                     SendBuff,
                     SendLen,
                     RecvBuff,
                     RecvLen,
                     &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending success
            }
```

5.1.5. SCardTransmit

The SCardTransmit function sends a service request to the smart card and expects to receive data back from the card.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379804%28v=vs.85%29.aspx

Note: *APDU Commands (i.e. the commands sent to connected card,* ***PCSC Pseudo APDU (with Proprietary Extension) for PICC****, and* ***Proprietary Pseudo APDU for PICC****) use this API for sending.*

Example:

```
#define SCARD_SCOPE_USER      0

SCARDCONTEXT      hContext;         // Resource manager context
SCARDHANDLE       hCard;            // Card context handle
unsigned long     dwActProtocol;    // Established active protocol
int               retCode;
char              readerName [256]; // List reader name
char              rName [256];      // Reader name for connect
BYTE              SendBuff[262],    // APDU command buffer
                  RecvBuff[262];    // APDU response buffer
BYTE              CardID [8],       // For storing the FeliCa IDM/
                                    MIFARE UID
BYTE              ResponseData[50]; // For storing card response
DWORD             SendLen,          // APDU command length
                  RecvLen;          // APDU response length
SCARD_IO_REQUEST  ioRequest;

void main ()
{
   …
   rName = "ACS ACR1552 1S CL Reader PICC 0"; // Depends on what
                                             reader should be used
                                             // Should connect to PICC
                                             interface

      retCode = SCardConnect(hContext,
                     rName,
                     SCARD_SHARE_SHARED,
                     SCARD_PROTOCOL_T0,
                     &hCard,
                     &dwActProtocol);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Connection failed (May be because of incorrect reader
         name, or no card was detected)
      }
      else
      {
         // Connection successful
         ioRequest.dwProtocol = dwActProtocol;
         ioRequest.cbPciLength = sizeof(SCARD_IO_REQUEST);
         RecvLen = 262;
```

```
// Get MIFARE UID/ FeliCa IDM
SendBuff[0] = 0xFF;
SendBuff[1] = 0xCA;
SendBuff[2] = 0x00;
SendBuff[3] = 0x00;
SendBuff[4] = 0x00;
SendLen = 5;
   retCode = SCardTransmit( hCard,
                  &ioRequest,
                  SendBuff,
                  SendLen,
                  NULL,
                  RecvBuff,
                  &RecvLen);

if (retCode != SCARD_S_SUCCESS)
{
   // APDU sending failed
   return;
}
else
{
   // APDU sending successful
   // The RecvBuff stores the IDM for FeliCa / the UID for
   MIFARE.
   // Copy the content for further FeliCa access
   for (int i=0;i< RecvLen-2;i++)
   {
      CardID [i] = RecvBuff[i];
   }
}
```

5.1.6. SCardDisconnect

The **SCardDisconnect** function terminates a connection previously opened between the calling application and a *smart card* in the target *reader*.

Refer to: http://msdn.microsoft.com/en-us/library/windows/desktop/aa379475%28v=vs.85%29.aspx

This function is used to end the PCSC Operation.

Example:

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT      hContext;         // Resource manager context
SCARDHANDLE       hCard;            // Card context handle
unsigned long     dwActProtocol;    // Established active protocol
int               retCode;

void main ()
{
   …
         // Connection successful
      …
      retCode = SCardDisconnect(hCard,SCARD_RESET_CARD);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Disconnection failed
      }
      else
      {
         // Disconnection successful
      }
   }
}
```

5.1.7. APDU Flow

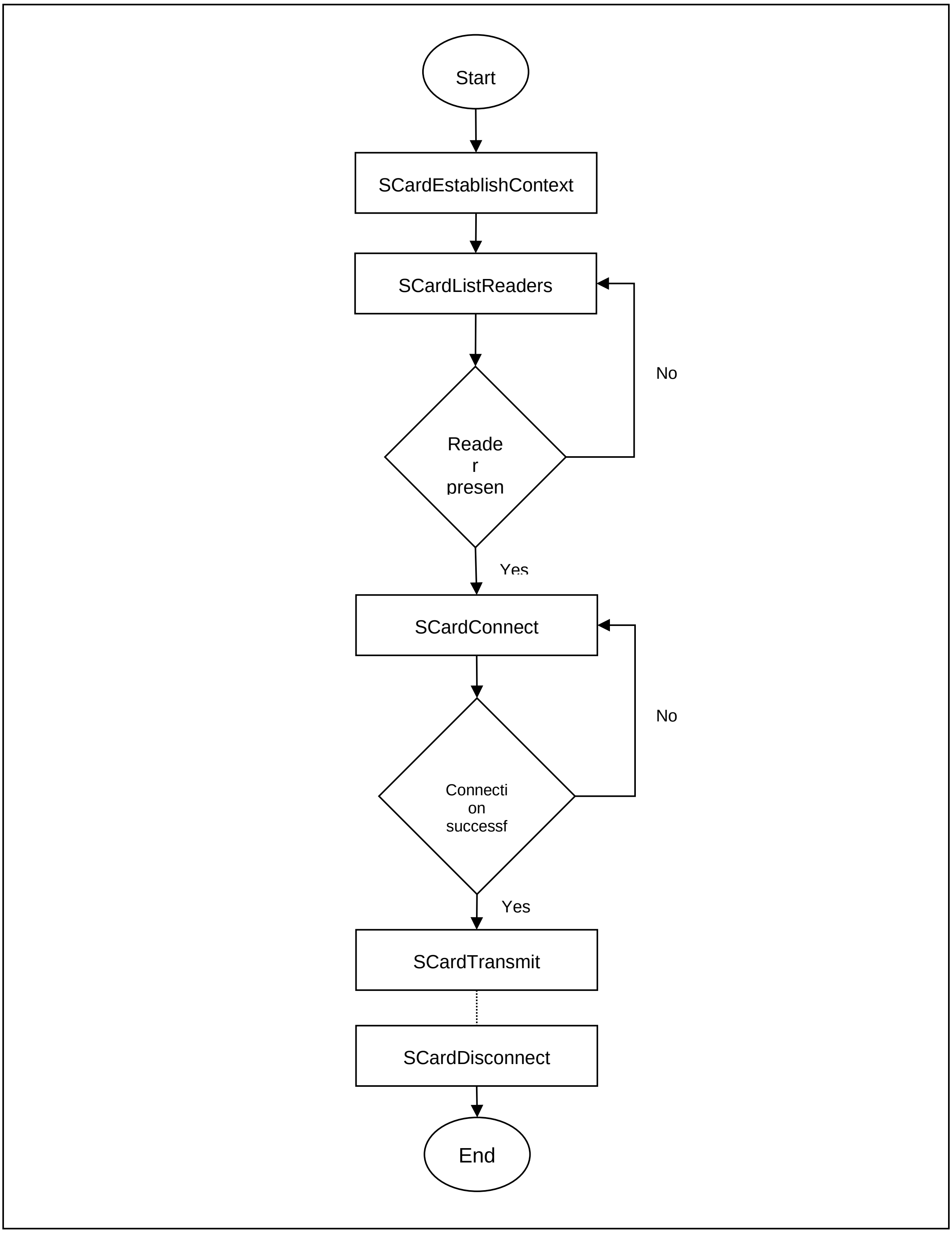


Figure 3: ACR1552U APDU Flow

5.1.8. Escape Command Flow

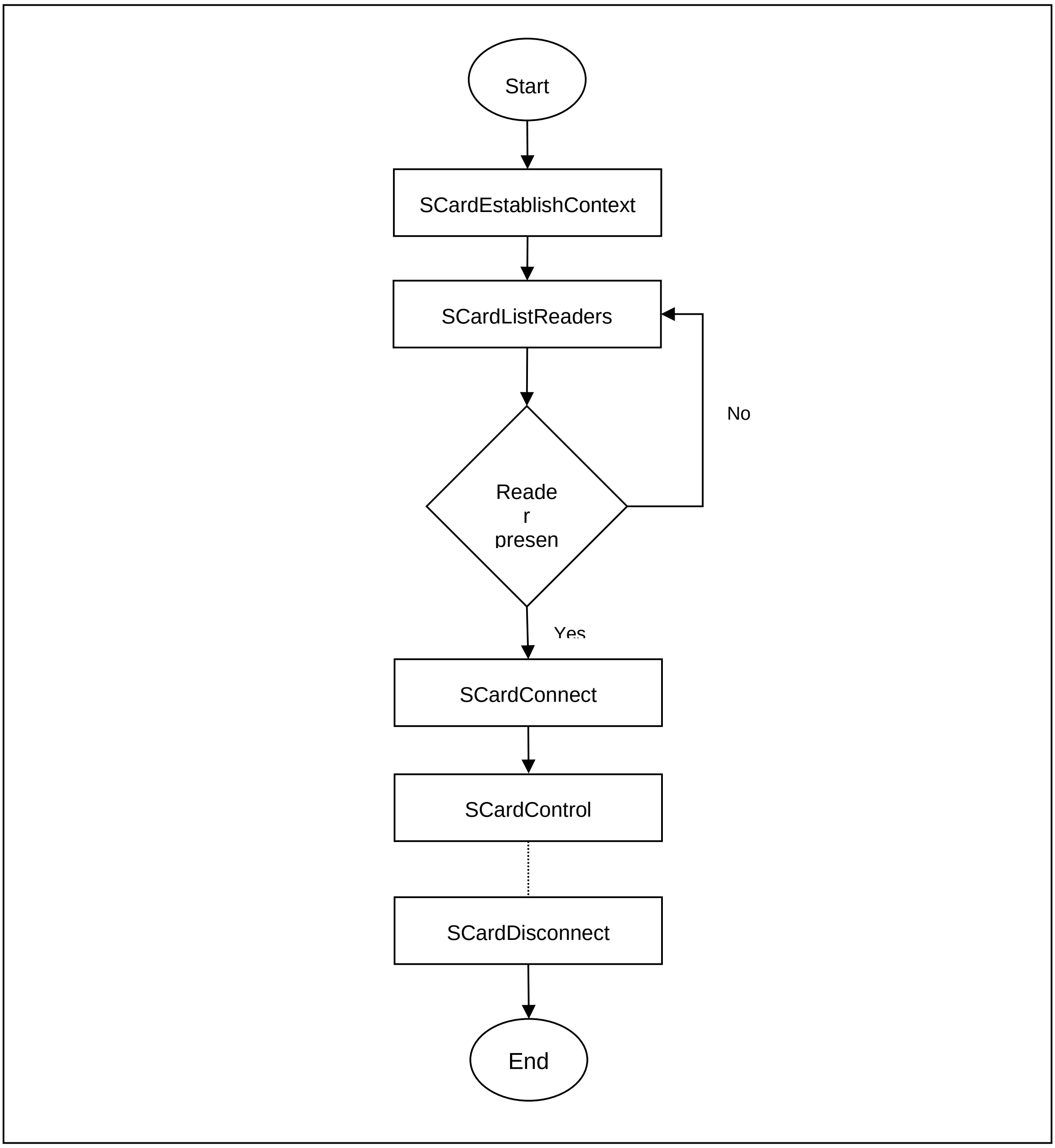


Figure 4: ACR1552U Escape Command Flow

5.2. Contact Smart Card Protocol

5.2.1. ACOS6-SAM Commands

This section contains SAM-specific commands. CCID Host could send Card Native Command or APDU to the Reader by using CCID Message PC_to_RDR_XfrBlock (corresponding to SCardTransmit() in PCSC API).

Note*: For complete information on ACOS6-SAM Commands and Scenarios, please contact an ACS representative for a copy of the ACOS6-SAM Reference Manual.*

5.2.1.1. Generate Key

This command is used to generate a diversified key to load into the ACOS3/6 card or other cards from deviation data such as a client card serial number. This command is catered for client card issuance purposes.

APDU	Description	
CLA	80h	
INS	88h	
P1	00h 01h 02h	Generate 8 Byte Key Generate 16 Byte Key Generate 24 Byte Key
P2	Key index of Master Key to generate Derived Key	
P3	08h	
Data	Input Data	

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3, must be 08h
6A 83h	Referenced key record not found in EF2
69 81h	Invalid EF2 (record size, file type, etc.)
6A 88h	EF2 not found
62 83h	Current DF is blocked; EF2 is blocked
69 83h	Usage counter is zero.
69 82h	Security condition not satisfied
6A 87h	Referenced Master Key is not capable of 3-DES encryption
61 08h	Command completed, issue GET RESPONSE to get the result

5.2.1.2. Diversify (or load) Key Data

This command prepares the SAM card to perform ciphering operations by diversifying and loading the key. It takes the serial number and CBC initial vector as command data input.

APDU	Description								
CLA	80h								
INS	72h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	0	0	1	Secret Code (Sc)
	-	0	0	0	0	0	1	0	Account Key (KACCT)
	-	0	0	0	0	0	1	1	Terminal Key
	-	0	0	0	0	1	0	0	Card Key
	-	0	0	0	0	1	0	1	Bulk Encryption Key (Not diversified)
	-	0	0	0	0	1	1	0	Initial vector
	0	-	-	-	-	-	-	-	16-byte Key
	1	-	-	-	-	-	-	-	24-byte Key
P2	Index of Master Key: Bit7: 1 = local Key in current EF2; 0 = global KEY EF2 Bit6-Bit5: 00b - RFU Bit4-Bit0: Key Index								
P3	If P1 = 1-4, P3 = 8/16,(if algo is AES, P3 = 8/16) If P1 = 5, P3 = 0 If P1 = 6, P3 = 8 (Algo of Master Key is DES/ 3DES/ 3KDES) P3 = 16 (Algo of Master Key is AES)								
Data	If P1 = 1-4 Client card's Serial Number, (if algo is AES, Data is Client card's Serial Number or Client card's Serial Number append with "0000000000000000") If P1 = 5, No command data. If P1 = 6, DES/3DES/3KDES/AES CBC initial vector.								

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Wrong P1, P1 must be 1 to 6
67 00h	Wrong P3, P3 must be 8 (or 0)
62 83h	Current DF is blocked, or EF2 is blocked
69 82h	Security condition not satisfied
6A 88h	EF2 not found
6A 83h	Referenced Master Key in EF2 not found

SW1 SW2	Description
69 81h	Invalid EF2 (FDB, MRL, etc., not consistent)
6A 87h	Referenced KEY not capable of authentication
69 83h	Referenced Key is locked
90 00h	Target key generated, and ready in SAM memory

5.2.1.3. Encrypt

This command is used to encrypt data using DES or 3DES with either:

1. The session key created by the mutual authentication procedure with an ACOS3/6, DESFire®, DESFire® EV1 or MIFARE Plus card.
2. A diversified key (secret code).
3. A bulk encryption key.
4. Encrypt the diversified secret code with the session key.
5. Prepare ACOS3 secure messaging command given a non-SM command.

APDU	Description								
CLA	80h								
INS	74h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	0	0	-	ECB Mode
	-	0	0	0	0	0	1	-	CBC Mode
	-	0	0	0	0	1	0	-	Retail MAC Mode
	-	0	0	0	0	1	1	-	MAC Mode
	-	0	0	0	1	0	0	-	Prepare ACOS3 SM command.
	-	1	0	0	1	0	1	-	MIFARE DESFire Encryption
	-	1	0	0	1	1	0	-	MIFARE DESFire EV1 Encryption
	-	0	0	0	1	1	1	-	CMAC
	-	0	1	0	0	0	0		MIFARE Plus Command
	-	0	1	0	0	0	1		MIFARE Plus Response
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES
	1	-	-	-	-	-	-	1	AES
	-	-	-	-	-	-	-	-	All other values – RFU

APDU	Description
P2	P2 is derived key in SAM set using Load Key function: 1 – Encrypt Data with Session Key *Ks* 2 – Encrypt Data with Diversified Key *Sc* 3 – Encrypt Data with Bulk Encryption Key 0 – return ENC (*Sc*, *Ks*) If P1.b3 = 1 or b5=1, P2 must be 1 If P2 = 0h, P1 can be either 0 or 1
P3	P3 < 128 If bit 3 of P1 not equal to 1 and bit 5 of P1 not equal to 1 - If P2 = 1-3, multiple of 8 (DES/3DES/3KDES) or 16 (AES) up to 128 bytes - If P2 = 0, 0
Data	Plain text If P2 b6 = 1, The DATA format should be: • Length of Plain text data • Length of Command and Header of DESFire Card • Command and Header of DESFire Card • Plain text P1 = A1h, the encryption is for a MIFARE Plus command • if MFP Command is *value* operations command, the DATA format should be Command code(1 BYTE)+BlockNum(2/4 BYTE)+Value(4 BYTE). • if MFP Command is *Proximity Check*, the DATA format should be Command code(1 BYTE)+ PPS1(1 BYTE). • if MFP Command is *Read*, the DATA format should be Command code(1 BYTE)+ BlockNum(2 BYTE) • if MFP Command is *Write*, the DATA format should be Command code(1 BYTE)+ BlockNum(2 BYTE) +plaintext P1=A3h, • The data return by ICC (don't include SC code and don't include RMAC if RMAC exist)

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3
6A 83h	ACOS Target Key is not ready (use Diversify to generate the key)
61 XX	Encryption is done, use GET RESPONSE to get the result

5.2.1.4. Decrypt

This command is used to decrypt data using DES or 3DES or AES with either:

1. The session key created by the mutual authentication procedure with an ACOS3/6, MIFARE DESFire, MIFARE DESFire EV1 or MIFARE Plus card.
2. A diversified key (secret code).
3. A bulk encryption key.
4. Decrypt the diversified secret code with the session key.
5. Verify and Decrypt ACOS3 secure-messaging response.

Verify and Decrypt ACOS3 SM Response:

APDU	Description								
CLA	80h								
INS	76h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	0	0	-	ECB Mode
	-	0	0	0	0	0	1	-	CBC Mode
	-	0	0	0	1	0	0	-	Verify and Decrypt ACOS3 SM Response
	-	1	0	0	1	0	1	-	MIFARE DESFire Decryption
	-	1	0	0	1	1	0	-	MIFARE DESFire EV1 Decryption
	-	0	1	0	0	1	0	-	MIFARE Plus Decryption
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES
	1	-	-	-	-	-	-	1	AES
	0	0	0	0	-	-	-	-	All other values - RFU
P2	P2 is derived key in SAM set using Load Key function: 1 – Decrypt Data with Session Key *Ks* 2 – Decrypt Data with Diversified Key *Sc* 3 – Decrypt Data with Bulk Encryption Key 0 – return DEC (*Sc*, *Ks*)								
P3	P3 < 128 If P1 = A5h, P3=16/32/48 If bit 3 of P1 not equal to 1 - If P2 = 1-3, multiple of 8 (DES/3DES/3KDES) or 16 (AES) up to 128 bytes - If P2 = 0, 0								
Data	Ciphertext If P1 = A5h, The DATA is Encrypted text If P2 b6 = 1, The DATA format should be: • Length of Plain text data, if unknown, use 00 • Length of Command and Header of DESFire Card • Command and Header of DESFire Card • Encrypted text								

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3
6A 83h	ACOS Target Key is not ready (use Diversify to generate the key)
61 XX	Decryption is done, use GET RESPONSE to get the result

5.2.1.5. Prepare Authentication

This command is used to authenticate the SAM card (as the terminal) to the ACOS 3/6 card or MIFARE Ultralight C/MIFARE DESFire Card/MIFARE Plus card.

APDU	Description
CLA	80h
INS	78h
P1	00h – 3DES 01h – DES 02h – 3KDES (MIFARE DESFire EV1/ACOS3) 03h – AES (MIFARE DESFire EV1/MIFARE Plus/ACOS3) 80h – 3DES (MIFARE DESFire Authenticate only) 81h – DES (MIFARE DESFire Authenticate only) Other – RFU
P2	0h – Verify ACOS3/6 Authenticate Return 01h – MIFARE Ultralight C/DESFire Authenticate by (Diversified) Terminal Key 05h – MIFARE Ultralight C/DESFire Authenticate by Bulk Encryption Key 02h – MIFARE Plus Authenticate. First Authenticate of SL1 to SL3 03h – MIFARE Plus Authenticate. Authentication in SL1 to SL2. 04h – MIFARE Plus Authenticate. Following Authenticate of SL2 to SL3.
P3	8 – (P1 = 00h, 01h, 02h, 80h, 81h) 16 – (P1 = 03h)
Data	Card Challenge Data

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3, must be 08h
6A 83h	ACOS Key (KT or KC) is not ready (use Diversify to generate this key)
69 82h	Security condition not satisfied
61 10h	Command completed, issue GET RESPONSE to get the result

5.2.1.6. Verify Authentication

This command is used to verify the ACOS 3/6, MIFARE Ultralight C, MIFARE DESFire/MIFARE DESFire EV1 or MIFARE Plus card to the terminal. The Session Key Ks would also be generated internally.

APDU	Description
CLA	80h
INS	7Ah
P1	00h – 3DES (P2 = 0) 01h – DES (P2 = 0) 02h – 3KDES (P2 = 0 ， ACOS3) 03h – AES (P2 = 0 ， ACOS3) Other – RFU
P2	00h – Verify ACOS3/6 Authenticate Return 01h – Verify MIFARE Ultralight C®/ DESFire®/ DESFire® EV1 Authenticate Return 02h – Verify MIFARE Plus Authenticate return
P3	08h – (P2 = 0, P2 = 1 and Session Key is DES/3DES) 16h – (P2 = 1 and Session Key is 3KDES/AES) 16h – (P2=02, and MIFARE Plus return data ek(RndA')) 32h – (P2=02, and MIFARE Plus return data ek(TI+PICCcap2+PCDcap2))
Data	ACOS 3/6: DES (K_S, RND_T) MIFARE DESFire/ DESFire EV1 return data: ek(RndA') MIFARE Plus return data ek(RndA') or ek(TI+PICCcap2+PCDcap2)

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3, must be 08h
6A 83h	ACOS-SAM Session Key or RND_T are not ready. Use PREPARE AUTHENTICATION to build these keys.
69 82h	Data is incorrect
90 00h	Data is correct, ACOS Mutual Authentication is successful

5.2.1.7. Verify ACOS Inquire Account

This command is used to verify the ACOS3/6 card's Inquire Account purse command. It would verify that the MAC checksum returned by ACOS3/6 are correct with the SAM's diversified key.

APDU	Description								
CLA	80h								
INS	7Ch								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	-	0	-	ACOS INQ_AUT is disabled
	-	0	0	0	0	-	1	-	ACOS INQ_AUT is enabled
	-	0	0	0	0	0	-	-	ACOS INQ_ACC_MAC is disabled
	-	0	0	0	0	1	-	-	ACOS INQ_ACC_MAC is enabled
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES (ACOS3 only)
	1	-	-	-	-	-	-	1	AES (ACOS3 only)
P2	0h								
P3	1Dh								
Data	Data Block returned by INQUIRE ACCOUNT of client ACOS card, see below.								

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3
6A 83h	ACOS Key K_S or K_{ACCT} are not ready; use DIVERSIFY command to generate K_{ACCT}; if applicable, use "Prepare Authentication" to generate K_S.
6F 00h	Data Block's MAC is incorrect
90 00h	Data Block's MAC is correct

5.2.1.8. Prepare ACOS Account Transaction

To create an ACOS3/6 Credit/Debit command, the MAC must be computed for ACOS3/6 to verify.

APDU	Description								
CLA	80h								
INS	7Eh								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	0	0	-	ACOS TRNS_AUT is disabled
	-	0	0	0	0	0	1	-	ACOS TRNS_AUT is enabled
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES (ACOS3 only)
	1	-	-	-	-	-	-	1	AES (ACOS3 only)
P2	E2h: Credit E6h: Debit								
P3	0Dh								
Data	Data Block								

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3, must be 0Dh
6A 83h	ACOS Key K_S or K_{ACCT} are not ready; use DIVERSIFY command to generate K_{ACCT}; if applicable, use “Prepare Authentication” to generate K_S.
61 0Bh	Command completed, issue GET RESPONSE to get the result

5.2.1.9. Verify Debit Certificate

For ACOS3/6, if the DEBIT command has P1 = 1, a debit certificate is returned. The debit certificate can be checked by comparing the ACOS3 response to the result of this command.

APDU	Description								
CLA	80h								
INS	70h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	Description
	-	0	0	0	0	0	0	-	ACOS TRNS_AUT is disabled
	-	0	0	0	0	0	1	-	ACOS TRNS_AUT is enabled
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES (ACOS3 only)

APDU	Description
	1 - - - - - - 1 AES (ACOS3 only)
P2	0h
P3	14h
Data	Data Block

Specific Response Status Bytes

SW1 SW2	Description
69 86h	No DF selected
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3, must be 14h
6A 83h	ACOS Key K_S or K_{ACCT} are not ready; use DIVERSIFY command to generate K_{ACCT}; if applicable, use PREPARE AUTHENTICATION to generate K_S.
69 82h	Security condition not satisfied
6F 00h	DEBIT CERTIFICATE is invalid
90 00h	Success, DEBIT CERTIFICATE is valid

5.2.1.10. Get Key

This command allows secure key injection from the current SAM’s Key File (SFI=02h) into another ACOS6/ACOS6-SAM with or without key diversification. Using this ensures that the keys to be injected are protected by encryption and message authentication codes.

The Get Key command also allows secure key injection from the current SAM’s Key File (SFI=02h) into ACOS7/10, MIFARE DESFire, MIFARE DESFire EV1 or MIFARE Plus card with key diversification. Using this ensures that the key to be injected is protected by encryption and message authentication codes.

If bit 7 of the Special Function Flag (Key Injection Only Flag) of the **Card Header Block** (Section 3.2 of ACOS6-SAM Reference Manual) has been set and the key file has been activated, Get Key must be used for loading or changing keys in the card. Setting this bit will disable Read Record command for the key file under any circumstances after activation.

Before this command is to be executed, a session key is already established with the target card with the mutual authentication procedure of **Mutual Authentication** (Section 5.3 of ACOS6-SAM Reference Manual) or the MIFARE Plus/MIFARE DESFire mutual authentication procedure.

Note: *The GET KEY command can only get the Key data.*

APDU	Description
CLA	80h
INS	CAh
P1	Get Key for ACOS card Set Key

APDU	Description			
	00h	Response data is Key in MSAM		
	01h	Response data is 16-byte Diversify Key		
	02h	Response data is 24-byte Diversify Key		
	03h	Response data is the Change Key command of MIFARE Plus Card		
	Get Key for DESFire card Change Key, Response data for DESFire/DESFire EV1 Change Key			
		Card Type	Authenticate Key No. And Changing Key No.*	Key Length
	80h	MIFARE DESFire	Are DIFFERENT in MIFARE DESFire card	16 bytes
	81h	MIFARE DESFire EV1	Are DIFFERENT in MIFARE DESFire EV1 card	16 bytes
	82h	MIFARE DESFire EV1	Are DIFFERENT in MIFARE DESFire EV1 card	24 bytes
	88h	MIFARE DESFire	Are the SAME in MIFARE DESFire card	16 bytes
	89h	MIFARE DESFire EV1	Are the SAME in MIFARE DESFire EV1 card	16 bytes
	8Ah	MIFARE DESFire EV1	Are the SAME in MIFARE DESFire EV1 card	24 bytes
P2	Key ID in SAM (New key for change)			
P3	If P1 = 00h, P3 is 08h If P1 = 01/02h, P3 is 10h If P1 = 03h, P3 is 0Bh If P1 = 80/81/82/88/89/8Ah: P3 is 0Bh			
Data	If P1 = 00h, command data is $\text{RND}_{\text{Target}}$ If P1 = 01/02h, command data is $\text{RND}_{\text{Target}}$ + serial (or batch) number of target card If P1 = 03h - Serial Number for target card (8 Byte) - Write Command (A0 or A1) (1 Byte) - BNr (2 Byte) If P1 = 80/81/82/88/89/8Ah: - Serial Number for target card (8 Byte) - Original Key ID (Key in SAM card stored the Original key, 00 = Default Key of DESFire - Card) - Key No. (DESFire Card Key No.) - Key Version (DESFire Card Key Version, If not used, value = 00)			

* This column points out if the listed cards have a distinct Change Key and Authenticate Key, or if they use the same value for both keys.

Specific Response Status Bytes

SW1 SW2	Description
69 85h	SAM Session Key not ready
62 83h	Current DF is blocked, or Target EF is blocked
69 86h	No DF selected
69 81h	Wrong file type of Key file, it should be Internal Linear Variable File
69 82h	Target file's header block has wrong checksum, or security condition not satisfied
6A 86h	Invalid P1 or P2
67 00h	Incorrect P3
6A 83h	Target Key is not ready or Key Length less than 16
61 1Ch	Success, use GET RESPONSE to get the result

5.3. Contactless Smart Card Protocol

5.3.1. ATR Generation

If the reader detects a PICC, an ATR will be sent to the PCSC driver for identifying the PICC.

5.3.1.1. ATR Format for ISO14443 Part 3 PICCs

Byte	Value	Designation	Description
0	3Bh	Initial Header	
1	8Nh	T0	Higher nibble 8 means: no TA1, TB1, TC1 only TD1 is following. Lower nibble N is the number of historical bytes (HistByte 0 to HistByte N-1)
2	80h	TD1	Higher nibble 8 means: no TA2, TB2, TC2 only TD2 is following. Lower nibble 0 means T = 0
3	01h	TD2	Higher nibble 0 means no TA3, TB3, TC3, TD3 following. Lower nibble 1 means T = 1
4 ~ 3+N	80h	T1	Category indicator byte, 80 means A status indicator may be present in an optional COMPACT-TLV data object
	4Fh	Tk	Application identifier Presence Indicator
	0Ch		Length
	RID		Registered Application Provider Identifier (RID) # A0 00 00 03 06
	SS		Byte for standard
	C0 .. C1h		Bytes for card name
	00 00 00 00h	RFU	RFU # 00 00 00 00
4+N	UU	TCK	Exclusive-oring of all the bytes T0 to Tk

Example:
ATR for MIFARE Classic 1K = {3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6Ah}
Where:

Length (YY) = 0Ch

RID = {A0 00 00 03 06h} (PC/SC Workgroup)
Standard (SS) = 03h (ISO 14443A, Part 3)
Card Name (C0 .. C1) = {00 01h} (MIFARE Classic 1K)
Standard (SS) = 03h: ISO 14443A, Part 3
= 11h: FeliCa

Card Name (C0 .. C1):

00 01: MIFARE Classic 1K
00 02: MIFARE Classic 4K
00 03: MIFARE Ultralight®
00 26: MIFARE Mini®
00 3A: MIFARE Ultralight® C
00 36: MIFARE Plus® SL1 2K
00 37: MIFARE Plus® SL1 4K
00 38: MIFARE Plus® SL2 2K
00 39: MIFARE Plus® SL2 4K
00 30: Topaz and Jewel
00 3B: FeliCa
FF 28: JCOP 30
FF [SAK]: undefined tags

5.3.1.2. ATR Format for ISO14443 Part 4 PICCs

<table>
<tr><th>Byte</th><th>Value</th><th>Designation</th><th>Description</th></tr>
<tr><td>0</td><td>3Bh</td><td>Initial Header</td><td></td></tr>
<tr><td>1</td><td>8Nh</td><td>T0</td><td>Higher nibble 8 means: no TA1, TB1, TC1 only TD1 is following.
Lower nibble N is the number of historical bytes (HistByte 0 to HistByte N-1)</td></tr>
<tr><td>2</td><td>80h</td><td>TD1</td><td>Higher nibble 8 means: no TA2, TB2, TC2 only TD2 is following.
Lower nibble 0 means T = 0</td></tr>
<tr><td>3</td><td>01h</td><td>TD2</td><td>Higher nibble 0 means no TA3, TB3, TC3, TD3 following.
Lower nibble 1 means T = 1</td></tr>
<tr><td rowspan="2">4 ~ 3+N</td><td>XX</td><td>T1</td><td rowspan="2">Historical Bytes:

ISO 14443-A:
The historical bytes from ATS response. Refer to the ISO 14443-4 specification.

ISO 14443-B:
<table>
<tr><th>Byte 1~4</th><th>Byte 5~7</th><th>Byte 8</th></tr>
<tr><td>Application Data from ATQB</td><td>Protocol Info Byte from ATQB</td><td>Higher nibble=MBLI from ATTRIB command Lower nibble (RFU)=0</td></tr>
</table></td></tr>
<tr><td>XX</td><td>Tk</td></tr>
<tr><td>4+N</td><td>UU</td><td>TCK</td><td>Exclusive-oring of all the bytes T0 to Tk</td></tr>
</table>

Example 1:
ATR for MIFARE® DESFire® = {3B 81 80 01 80 80h} // 6 bytes of ATR

***Note:** Use the APDU “FF CA 01 00 00h” to distinguish the ISO 14443A-4 and ISO 14443B-4 PICCs, and retrieve the full ATS if available. ISO 14443A-3 or ISO 14443B-3/4 PICCs do have ATS returned.*

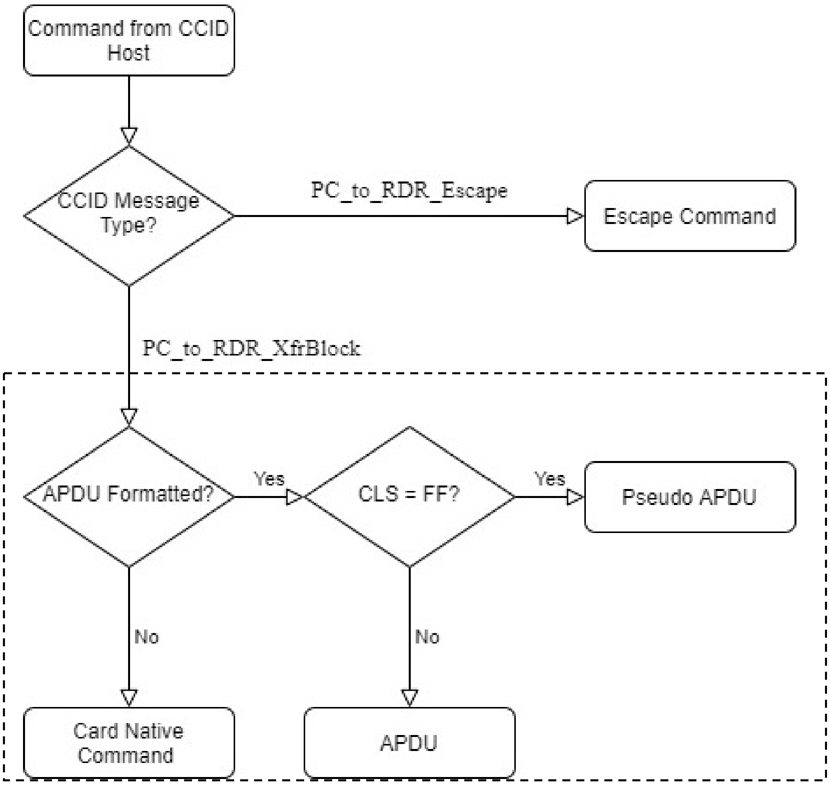
APDU Command	= FF CA 01 00 00h
APDU Response	= 06 75 77 81 02 80 90 00h
ATS	= {06 75 77 81 02 80h}

Example 2:

ATR for EZ-Link	= {3B 88 80 01 1C 2D 94 11 F7 71 85 00 BEh}
Application Data of ATQB	= 1C 2D 94 11h
Protocol Information of ATQB	= F7 71 85h
MBLI of ATTRIB	= 00h

5.3.2. APDU, Pseudo APDU and Card Native Command

User should use PC_to_RDR_XfrBlock Message to send APDU, Pseudo APDU and Card Native Command to the reader. After the command processing, the Reader will send back the response by RDR_to_PC_DataBlock Message.



CCID Host could send Card Native Command or APDU to the Reader by using CCID Message PC_to_RDR_XfrBlock (corresponding to SCardTransmit() in PCSC API). For PICC, if the card support ISO14443 part 4 protocol or Innovatron protocol, the Reader will pack the Command/APDU into the protocol frame and send to the card directly without any interpretation of the Command/APDU. If the card do not support neither protocol, a message “6A 81” will return to CCID Host.

Note: Due to Microsoft Window Smart Card Plug and Play, Microsoft Window may send some APDU to a card at the time of card present. This action will make a DESFire card entering ISO APDU mode such that the card become fail to receive a native command until a card reset. Usually Microsoft Window will reset the card (by PC_to_RDR_IccPowerOff) after 10s of inactive.

5.3.3. PCSC Pseudo APDU (with Proprietary Extension) for PICC

The following Pseudo APDUs are provided to access a contactless card indirectly. CCID Host could send these APDUs to Reader by using CCID Message PC_to_RDR_XfrBlock (corresponding to SCardTransmit() in PCSC API). After receiving of a Pseudo APDU, it will be interpreted to generate low level card command(s) and then send to card. After the card handling those low level command(s), Reader collect the response(s) from the card and create a response to send back to CCID Host.

5.3.3.1. Get Data [FF CA …]

This command is used to read out the data obtained during activation process, such as serial number, protocol parameter etc.

Command

Command	Class	INS	P1	P2	Le
Get Data	FFh	CAh	See below		00h (Full Length)

Command Parameter

P1	P2	Meaning
00h	00h	Get the UID/PUPI/SN of the Card
01h	00h	Get the ATS for Type A Part 4
00h	02h	Get the following Card Type related data in transmission order: Type A: 2 bytes ATQA/ATVA + 4/7/10 Bytes UID + 1 bytes Last SAK. Type B: 12 bytes ATQB
80h	00h	Get the following Card Type related data in transmission order: Type A: 2 bytes ATQA/ATVA + 4/7/10 Bytes UID + 1/2/3 bytes SAK. Type B: 12 bytes ATQB FeliCa: 17 byte ATQ (+ 6 byte ATTR if activated) SRI: 8 byte UID + 1 byte Chip ID. ISO15693: 1 byte DSFID + 8 byte UID CTS: 4 byte SN + 2 byte ATQT Innovatron: 4 byte SN + 1 byte tag address.

Response

Response	Data Out		
Result	Data	SW1	SW2

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Examples:

To get the serial number of the "connected PICC":

```
UINT8 GET_UID[5] = {FF, CA, 00, 00, 00};
```

To get the ATS of the “connected ISO 14443 A PICC”:

UINT8 GET_ATS[5] = {FF, CA, 01, 00, 00};

5.3.3.2. Load Key [FF 82 …]

This command is used to set the Key Data to the internal key buffer specified by Key Buffer Number. The key buffer is volatile and its content would be used during authentication. This command will not generate card data transfer.

Command

Command	Class	INS	P1	P2	Lc	Data In
Load Authentication Keys	FFh	82h	00h	Key Buffer Number (0 to 1)	Key Length	Key Data

Key Length/Data

Card Type	Key Length (Lc)	Key Data (in Transmission/Storing Order)
MIFARE Standard MIFARE Plus SL1	06h	6 Bytes Crypto1 Key A/B.
MIFARE Plus SL1 MIFARE Plus SL2	16h	6 Bytes Crypto1 Key A/B + 16 Bytes AES Key.
MIFARE Plus SL2	06h	6 Bytes Encrypted Crypto1 Key A/B.
MIFARE UltraLightC MIFARE DESFire	10h	16 Bytes 2K3DES Key.

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Example:

// Load a key {FF FF FF FF FF FFh} into the volatile memory location 00h.

APDU = {FF 82 00 00 06 FF FF FF FF FF FFh}

5.3.3.3. Authenticate [FF 86 00 00 05 …]

This command is used to performing an authentication to the card to grand access of the protected blocks/pages. Before sending this command, User should use Load Key command to set the correct key data to the buffer specified by **Key Buffer Number**.

Command

Command	Class	INS	P1	P2	Lc	Data In
Authenticate	FFh	86h	00h	00h	05h	See Below

Command Data

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
01h	00h (RFU)	Address	Key Type	Key Buffer Number

Address and Key Type

Card Type	Address	Key Type
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	00h~FFh: Block 0~255	60h: Crypto1 Key A 61h: Crypto1 Key B
MIFARE UltraLightC	00h (RFU)	80h: 2K3DES
MIFARE DESFire	00h~0Eh: DESFire Key Number 0~14	0Ah: 2K3DES

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Sectors (Total 16 sectors. Each sector consists of 4 consecutive blocks)	Data Blocks (3 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)
Sector 0	00h – 02h	03h
Sector 1	04h – 06h	07h
..	..	..
..	..	..
Sector 14	38h – 0Ah	3Bh
Sector 15	3Ch – 3Eh	3Fh

1 KB

Table 3: MIFARE Classic 1K Memory Map

Sectors **(Total 32 sectors. Each sector consists of 4 consecutive blocks)**	**Data Blocks** **(3 blocks, 16 bytes per block)**	**Trailer Block** **(1 block, 16 bytes)**
Sector 0	00h ~ 02h	03h
Sector 1	04h ~ 06h	07h
..		
..		
Sector 30	78h ~ 7Ah	7Bh
Sector 31	7Ch ~ 7Eh	7Fh

Sectors **(Total 8 sectors. Each sector consists of 16 consecutive blocks)**	**Data Blocks** **(15 blocks, 16 bytes per block)**	**Trailer Block** **(1 block, 16 bytes)**
Sector 32	80h ~ 8Eh	8Fh
Sector 33	90h ~ 9Eh	9Fh
..		
..		
Sector 38	E0h ~ EEh	EFh
Sector 39	F0h ~ FEh	FFh

2 KB

Table 4: MIFARE Classic 4K Memory Map

Byte Number	0	1	2	3	Page
Serial Number	SN0	SN1	SN2	BCC0	0
Serial Number	SN3	SN4	SN5	SN6	1
Internal/Lock	BCC1	Internal	Lock0	Lock1	2
OTP	OPT0	OPT1	OTP2	OTP3	3
Data read/write	Data0	Data1	Data2	Data3	4
Data read/write	Data4	Data5	Data6	Data7	5
Data read/write	Data8	Data9	Data10	Data11	6
Data read/write	Data12	Data13	Data14	Data15	7
Data read/write	Data16	Data17	Data18	Data19	8
Data read/write	Data20	Data21	Data22	Data23	9
Data read/write	Data24	Data25	Data26	Data27	10
Data read/write	Data28	Data29	Data30	Data31	11
Data read/write	Data32	Data33	Data34	Data35	12
Data read/write	Data36	Data37	Data38	Data39	13
Data read/write	Data40	Data41	Data42	Data43	14
Data read/write	Data44	Data45	Data46	Data47	15

512 bits
or
64 bytes

Table 5: MIFARE Ultralight Memory Map

Examples:

// To authenticate the Block 04h with a {TYPE A, key number 00h}. PC/SC V2.01, Obsolete

APDU = {FF 88 00 04 60 00h};

// To authenticate the Block 04h with a {TYPE A, key number 00h}. PC/SC V2.07

APDU = {FF 86 00 00 05 01 00 04 60 00h}

***Note:** MIFARE Ultralight does not need to do any authentication. The memory is free to access.*

5.3.3.4. Read Binary Blocks [FF B0 …]

This command is used to read specified number of byte of data from PICC starting from the specified block/page address. Depend on card type, user may need to perform authentication to get the access right of the required block(s)/page(s) before sending this command.

Command:

Command	Class	INS	P1	P2	Le
Read Binary Blocks	FFh	B0h	Mode and Address		Number of Bytes to Read

P1/P2 (Mode and Address)

Card Type	P1[7:4] Mode	P1[3:0] + P2[7:0] Starting Address (MSB First)
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	00h: Skip Trailers 08h: With Trailers	000h~0FFh: Block 0~255
MIFARE UltraLight MIFARE UltraLightC	00h (Reserved)	000h~02Fh: Page 0~47
SRIX4K/SRT512	00h (Reserved)	000h~07Fh: Block 0~127 0FFh: System Area
PicoPass	00h (Reserved)	000h~0FFh: Block 0~255
Topaz/NFC Type-1 Tag	00h (Reserved)	000h~7FFh: Byte Address

Le (Number of Bytes to Read)

Type	Byte 0	Byte 1	Byte 2
Short	00h: Read 256 bytes 01h~FFh: Read 1~255 bytes	--	
Extended	00h	0000h: Read 65536 bytes 0001h~FFFFh: Read 1~65535 bytes	

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Examples:

// Read 16 bytes from the binary block 04h (MIFARE Classic 1K or 4K)

APDU = FF B0 00 04 10h

// Read 240 bytes starting from the binary block 80h (MIFARE Classic 4K)

// Block 80h to Block 8Eh (15 blocks)

APDU = FF B0 00 80 F0h

5.3.3.5. Update Binary Blocks [FF D6 …]

This command is used to write specified number (must be multiple of block/page size) of bytes to PICC starting from the specified block/page address. Depend on card type, user may need to perform authentication to get the access right of the required block(s)/page(s) before sending this command.

User should take a great care for writing to block/page that may change the security setting of the card (e.g. sector trailers of MIFARE card) as this may lock the card if incorrect data is written or operation is failed. As a result, to minimize the risk of card locking, it is not recommended to write to multiple block/page in a single APDU command if security block/page is involved.

Command

Command	Class	INS	P1	P2	Lc	Data In
Update Binary Blocks	FFh	D6h	Mode and Address		Number of Bytes to Write	Data Bytes

P1/P2 (Mode and Address) and Write Size alignment (Block/Page Size)

Card Type	P1[7:4] Mode	P1[3:0] + P2[7:0] Starting Address (MSB First)	Blk/Page Size (Bytes)
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	0x0: Skip Trailers 0x8: With Trailers	000h~0FFh: Block 0~255	16
MIFARE UltraLight MIFARE UltraLightC	0x0 (Reserved)	000h~02Fh: Page 0~47	4
SRIX4K/SRT512	0x0 (Reserved)	SRIX4K/SRT512	4
PicoPass	0x0 (Reserved)	PicoPass	8
Topaz/NFC Type-1 Tag	0x0: with Erase	000h~7FFh: Byte Address	1(Addr 78h) or 8(Else)
	0x8: without Erase		

Lc (Number of Bytes to Write)

Type	Byte 0	Byte 1	Byte 2
Short	01h~FFh: Write 1~255 bytes	--	
Extended	00h	0001h~FFFFh: Write 1~65535 bytes	

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Examples:

// Update the binary block 04h of MIFARE Classic 1K/4K with Data {00 01 .. 0Fh}

APDU = {FF D6 00 04 10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0Fh}

// Update the binary block 04h of MIFARE Ultralight with Data {00 01 02 03h}

APDU = {FF D6 00 04 04 00 01 02 03h}

5.3.3.6. APDU Commands for PCSC 2.0 Part 3 (Version 2.02 or above)

PCSC2.0 Part 3 commands are used to transparently pass data from an application to a contactless tag, return the received data transparently to the application and protocol, and switch the protocol simultaneously.

5.3.3.6.1. Command and Response APDU Format

Command Format

CLA	INS	P1	P2	Lc	Data In
FFh	C2h	00h	Function	DataLen	Data[DataLen]

Where Functions (1 byte)**:**

00h = Manage Session
01h = Transparent Exchange
02h = Switch Protocol
Other = RFU

Response Format

Data Out	SW1	SW2
Data Field BER-TLV encoded		

Every command returns SW1 and SW2 together with the response data field (if available). The SW1 SW2 is based on ISO 7816. SW1 SW2 from the C0 data object below should also be used.

C0 data element Format

Tag	Length (1 byte)	SW2
C0h	03h	Error Status

Error Status Description

Error Status	Description
XX SW1 SW2	XX = number of the bad data object in the APDU 00 = general error of APDU 01 = error in the 1st data object 02 = error in the 2nd data object
00 90 00h	No error occurred
XX 62 82h	Data object XX warning, requested information not available
XX 63 00h	No information
XX 63 01h	Execution stopped due to failure in other data object
XX 6A 81h	Data object XX not supported
XX 67 00h	Data object XX with unexpected length
XX 6A 80h	Data object XX with unexpected value
XX 64 00h	Data Object XX execution error (no response from IFD)
XX 64 01h	Data Object XX execution error (no response from ICC)
XX 6F 00h	Data object XX failed, no precise diagnosis

The first value byte indicates the number of the erroneous data object XX, while the last two bytes indicate the explanation of the error. SW1 SW2 values based on ISO 7816 are allowed.

If there are more than one data objects in the C-APDU field and one data object failed, IFD can process the following data objects if they do not depend on the failed data objects.

5.3.3.6.2. Manage Session [FF C2 00 00 …]

This command allows user to start a session with polling disable for the following communication. User should end the session as soon as those communications finished.

Please note, this command may make the reader fail detect a card present/absence if used incorrectly. This fail may be unable to recover automatically until a logical/physical reader disconnection.

Command

Command	Class	INS	P1	P2	Lc	Data In	Le
Manage Session	FFh	C2h	00h	00h	Cmd Data Length	Cmd TLV	--/00h

Response Code

Rsp Data	SW1 SW2	Meaning
--	90 00h	The operation was completed successfully.
Rsp TLV	90 00h	For Le = 0x00, One of Command TLV Fail. For Detail of Error, refer to Rsp TLV.
--	6X XXh	For Le = --, One of Command TLV Fail.

Cmd TLV

Cmd	Meaning
Start Session: 81 00h	Start a Session and Disable Polling.
RF Off: 83 00h	Turn off RF.
Timer: 5F 46 04h [TIME]	Set the sleep time before the next RF On/Off TLV. [TIME]: 4 byte value (MSB first) in range from 1000 to 100000 us. The actual sleep time will round up to nearest 1000us.
RF On: 84 00h	Turn on RF.
End Session: 82 00h	End a Session and Re-enable Polling.

Rsp TLV

Rsp	Meaning
TLV Error: C0 03 NN 6X XXh	Error in the NNth Command TLV.

5.3.3.6.2.1. Start Session Data Object

This command is used to start a transparent session. Once the session has started, auto-polling will be disabled until the session is ended.

Start Session Data Object

Tag	Length (1 byte)	Value
81h	00h	-

5.3.3.6.2.2. End Session Data Object

This command ends the transparent session. The auto-polling will be reset to the state before the session has started.

End Session Data Object

Tag	Length (1 byte)	Value
82h	00h	-

5.3.3.6.2.3. Turn Off the RF Data Object

This command turns off the antenna field.

Turn off RF Field Data Object

Tag	Length (1 byte)	Value
83h	00h	-

5.3.3.6.2.4. Turn On the RF Data Object

This command turns on the antenna field.

Turn on the RF Field Data Object

Tag	Length (1 byte)	Value
84h	00h	-

5.3.3.6.2.5. Timer Data Object

This command creates a 32-bit timer data object in unit of 1 µs.

Example: If there is a timer data object with 5000 µs between RF Turn Off Data Object and RF Turn On Data Object, the reader will turn off the RF field for about 5000µs before it is turned on.

Timer Data Object

Tag	Length (1 byte)	Value
5F 46h	04h	Timer (4 bytes)

5.3.3.6.3. Transparent Exchange [FF C2 00 01 …]

This command allows user transmit and receive any bit or bytes to/from card, with option to configure various link and transport layer (e.g. ISO14443 part 4) and some link layer redundancy (CRC and parity) optionally. User could embed any card specific raw data into this pseudo APDU and then send to the card.

Please note, this command may interference internal handling of card support, may change the card status without notification to the driver/firmware and may require a card reset and/or removal to bring the driver/firmware back to normal.

Command

Command	Class	INS	P1	P2	Lc	Data In	Le
Transparent Exchange	FFh	C2h	00h	01h	Cmd Data Length	Cmd TLV	00h

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

Cmd TLV

Cmd	Meaning
Transceive Flag: 90 02 [Flag] 00h	Set the Flag for the following Transceive TLV. Flag[7:5]: RFU; Set to 0 Flag[4]: Set to disable ISO14443 Part 4 Flag[3]: Set to disable receiving parity handling Flag[2]: Set to disable transmitting parity handling Flag[1]: Set to disable receiving CRC handling Flag[0]: Set to disable transmitting CRC handling If this TLV is missing, the Flag value set in previous command is used. If Flag value is never set, current protocol value is used.
Transmit Bit Frame: 91 01h [NumBit]	Set the Bit Frame for the following Transceive TLV. If this TLV is missing, the default value is 0. NumBit[7:3]: RFU; Set to 0 NumBit[2:0]: Number of valid bits in last byte (0 means all valid).
Timer: 5F 46 04h [TIME]	Set the timeout for the following Transceive TLV. [TIME]: 4 byte value (MSB first) in range 1 us to 1000000 us. The actual timeout will round up to nearest 302.07 x 20~15 us. If this TLV is missing, the FWTI value set previously will be used as timeout.
Set FWTI:	Set FWT/Timeout for Transceive. If FWTI does not set by any previous “FF C2h ...” command, the default value is 0.

Cmd	Meaning
FF 6E 03 03 01h [FWTI]	FWTI: 0 ~ 15, FWT/Timeout = 302.07 x 2FWTI us
Transceive: 95h [Size] [Data]	Size: Size of Data coded in BER-TLV length field. Data: Data to be Transmit.

Rsp TLV

Rsp	Meaning
Receive Bit framing: 92 01h [NumBit]	NumBit[7:3]: RFU; Set to 0. NumBit[2:0]: Number of valid bits in last byte (0 means all valid).
Response: 97h [Size] [Data]	Size: Size of Data coded in BER-TLV length field. Data: Data Received.
Response Status: 96 02h [Status] 00h	Status [7:4]: RFU. Status[3]: Framing Error. Status[2]: Parity Error. Status[1]: RFU. Status[0]: CRC Error.

5.3.3.6.3.1. Transmission and Reception Flag Data Object

This command defines the framing and RF parameters for the following transmission.

Transmission and Reception Flag Data Object

Tag	Length (1 byte)	Value		
		Byte 0		Byte 1
		bit	Description	
90h	02h	0	0 – append CRC in the transmit data 1 – do not append CRC in the transmit data	00h
		1	0 – CRC checking from the received data 1 – no CRC checking from the received data	
		2	0 – insert parity in the transmit data 1 – do not insert parity	
		3	0 – expect parity in received date 1 – do not expect parity (i.e. no parity checking)	
		4	0 – append protocol prologue in the transmit data or discard from the response 1 – do not append or discard protocol prologue if any (e.g. PCB, CID, NAD)	
		5-7	RFU	

5.3.3.6.3.2. Transmission Bit Framing Data Object

This command defines the number of valid bits of the last byte of data to transmit or transceive.

Transmission bit Framing Data Object

Tag	Length (1 byte)	Value	
		bit	Description
91h	01h	0-2	Number of valid bits of the last byte (0 means all bits are valid)
		3-7	RFU

Transmission bit framing data object shall be together with “transmit” or “transceive” data object only. If this data object does not exist, it means all bits are valid.

5.3.3.6.3.3. Transceive Data Object

This command transmits and receives data from the ICC. After transmission is complete, the reader will wait until the time given in the timer data object.

If no timer data object was defined in the data field, the reader will wait for the duration given in the Set Parameter FWTI Data Object. If no FWTI is set, the reader will wait for about 302 µs.

Transceive Data Object

Tag	Length (1 byte)	Value
95h	DataLen	Data (N Bytes)

5.3.3.6.3.4. Timer Data Object

This command creates a 32-bit timer data object in unit of 1 µs.

Example: If there is a timer data object with 5000 µs, the reader will wait the following Transceive TLV for about 5000µs before timeout.

Timer Data Object

Tag	Length (1 byte)	Value
5F 46h	04h	Timer (4 bytes)

5.3.3.6.3.5. Response Bit Framing Data Object

Inside the response, this command is used to notify the received transmission bit Framing Data Object

Tag	Length (1 byte)	Value	
		bit	Description
92h	01h	0-2	Number of valid bits of the last byte (0 means all bits are valid)
		3-7	RFU

Transmission bit framing data object shall be together with “transmit” or “transceive” data object only. If this data object does not exist, it means all bits are valid.

5.3.3.6.3.6. Response Status Data Object

Inside the response, this command is used to notify the received data status.

Response Status Data Object

Tag	Length (1 byte)	Value		
		Byte 0		Byte 1
		Bit	Description	
96h	02h	0	0 – CRC is OK or no checked 1 – CRC check fail	RFU
		1	0 – no collision 1 – collision detected	
		2	0 – no parity error 1 – parity error detected	
		3	0 – no framing error 1 – framing error detected	
		4 - 7	RFU	

5.3.3.6.3.7. Response Data Object

Inside the response, this command is used to notify the received data status.

Response Data Object

Tag	Length (1 byte)	Value
97h	DataLen	ReplyData (N Byte)

5.3.3.6.4. Switch Protocol [FF C2 00 02 …]

This command allows user to switch to specify protocol, select protocol layer and parameter.

Please note, this command may interference internal handling of card support, may change the card status without notification to the driver/firmware and may require a card reset and/or removal to bring the driver/firmware back to normal.

Command

Command	Class	INS	P1	P2	Lc	Data In	Le
Switch Protocol	FFh	C2h	00h	02h	Cmd Data Length	Cmd TLV	00h

Response Code

Rsp Data	SW1 SW2	Meaning
Rsp TLV	90 00h	Succeed with data.
--	90 00h	Succeed.
--	6X XXh	Fail.

Cmd TLV

Cmd	Meaning
Set Baud: FF 6E 03 05 01h [Baud]	Set the Baud for Part/Layer 4 to be applied during Switch Protocol. If [Baud] does not set by any previous "FF C2h ..." command, the default value is 98h (106 kbps). Baud[7:2]: RFU, Set to 100110b. Baud[1:0]: Baud to be set, 00b (106 kbps), 01b (212 kbps), 10b (424 kbps), 11b (848 kbps).
Switch Protocol: 8F 02h [RF] [Layer]	Switch the protocol to specified RF and/or Layer. [RF]: 00h: ISO14443A, 01h: ISO14443B 02h: ISO15693, 03h: FeliCa, FFh: Current RF Other: RFU [Layer]: 02h: Layer/Part 2, 03h: Layer/Part 3, 04h: Layer/Part 4 (For A/B Only) Other: RFU Note: It must be in a Transparent Session (Disable Polling) if switching to Layer/Part 2.

Rsp TLV

Rsp	Meaning
Response: 8Fh [Size] [Data]	Size: Size of Data coded in BER-TLV length field. Data: ATR (if Part 4) or Final SAK (if Type A part 3) or PI in ATQB (if Type B part 3).

5.3.3.6.4.1. Switch Protocol Data Object

This command specifies the protocol and different layers of the standard.

Switch Protocol Data Object

Tag	Length (1 byte)	Value	
		Byte 0	Byte 1
8Fh	02h	00h – ISO/IEC14443 Type A 01h – ISO/IEC14443 Type B 02h – ISO15693 03h – FeliCa Other – RFU	02h – Switch to Layer 2 03h – Switch or activate to layer 3 04h – Activate to layer 4 Other - RFU

5.3.3.6.4.2. Response Data Object

Inside the response, this command is used to notify the received data status.

Response Data Object

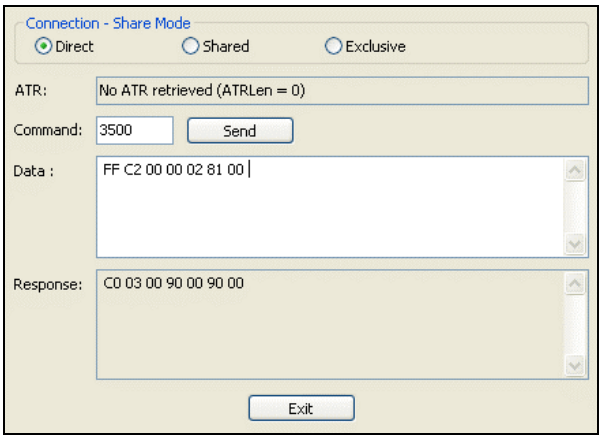
Tag	Length (1 byte)	Value
5F 51h	DataLen	ATR
8Fh	DataLen	Final SAK (if Type A part 3) or PI in ATQB (if Type B part 3).

5.3.3.6.5. PCSC 2.0 Part 3 Example

1. Start Transparent Session.

   Command: **FF C2 00 00 02 81 00**

   Response: **C0 03 00 90 00 90 00**



ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

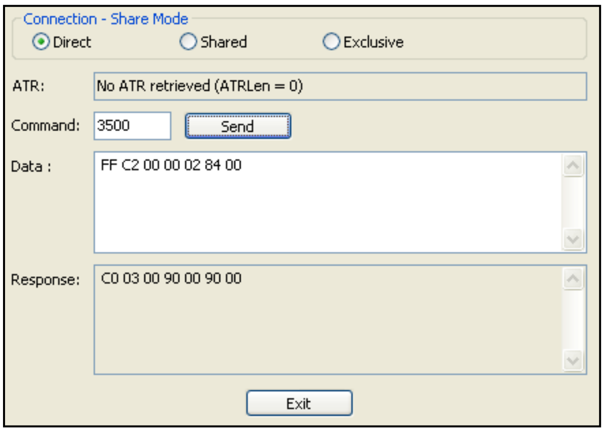
Data : FF C2 00 00 02 81 00

Response: C0 03 00 90 00 90 00

2. Turn the Antenna Field on.

   Command: **FF C2 00 00 02 84 00**

   Response: **C0 03 00 90 00 90 00**



ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

Data : FF C2 00 00 02 84 00

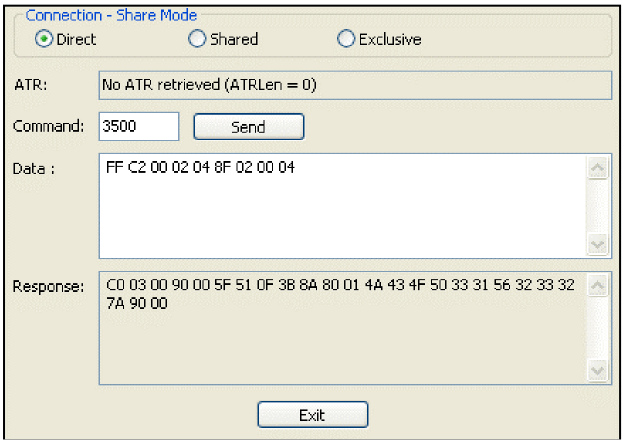
Response: C0 03 00 90 00 90 00

3. ISO 14443-4A Active.

   Command: **FF C2 00 02 04 8F 02 00 04**

   Response: **C0 03 01 64 01 90 00** (if no card present)

   C0 03 00 90 00 5F 51 [Len] [ATR] 90 00



Connection - Share Mode

● Direct ○ Shared ○ Exclusive

ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

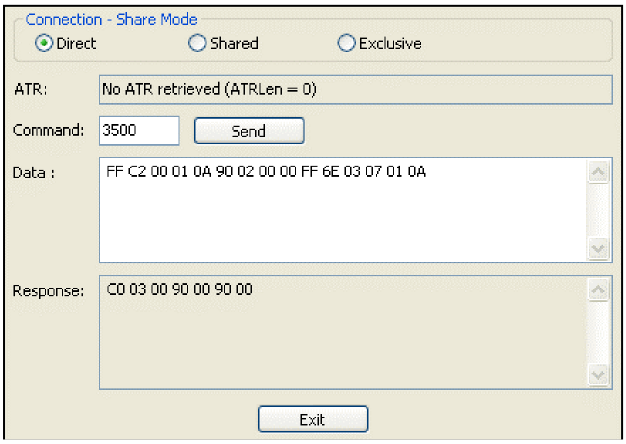
Data : FF C2 00 02 04 8F 02 00 04

Response: C0 03 00 90 00 5F 51 0F 3B 8A 80 01 4A 43 4F 50 33 31 56 32 33 32 7A 90 00

4. Set the PCB to 0Ah and enable the CRC, parity and protocol prologue in the transmit data.

   Command: **FF C2 00 01 0A 90 02 00 00 FF 6E 03 07 01 0A**

   Response: **C0 03 00 90 00 90 00**



Connection - Share Mode

● Direct ○ Shared ○ Exclusive

ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

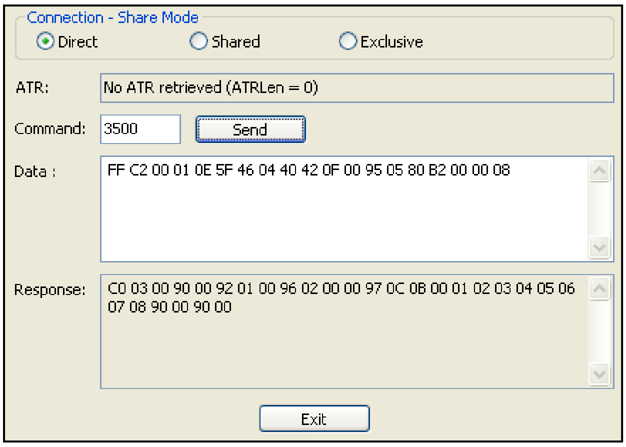
Data : FF C2 00 01 0A 90 02 00 00 FF 6E 03 07 01 0A

Response: C0 03 00 90 00 90 00

5. Send the APDU “80B2000008” to card and get response.

Command: **FF C2 00 01 0E 5F 46 04 40 42 0F 00 95 05 80 B2 00 00 08**

Response: **C0 03 00 90 00 92 01 00 96 02 00 00 97 0C [Card Response] 90 00**



Connection - Share Mode
● Direct ○ Shared ○ Exclusive

ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

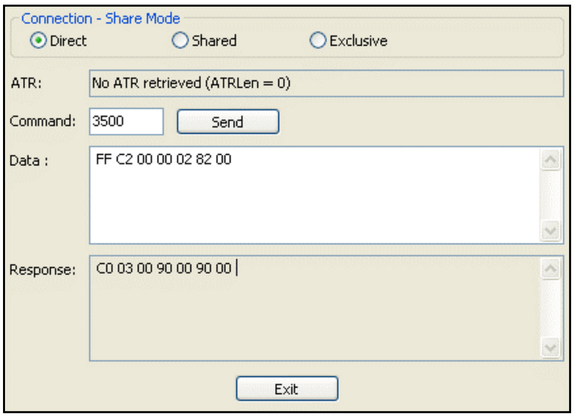
Data : FF C2 00 01 0E 5F 46 04 40 42 0F 00 95 05 80 B2 00 00 08

Response: C0 03 00 90 00 92 01 00 96 02 00 00 97 0C 0B 00 01 02 03 04 05 06 07 08 90 00 90 00

6. End Transparent Session.

Command: **FF C2 00 00 02 82 00**

Response: **C0 03 00 90 00 90 00**



Connection - Share Mode
● Direct ○ Shared ○ Exclusive

ATR: No ATR retrieved (ATRLen = 0)

Command: 3500

Data : FF C2 00 00 02 82 00

Response: C0 03 00 90 00 90 00

5.3.4. Proprietary Pseudo APDU for PICC

The following Pseudo APDUs are provided as supplement to PCSC Pseudo APDUs to access a contactless card indirectly. The internally handling of these APDU is similar to PCSC Pseudo APDUs.

5.3.4.1. Write Value Block [FF D7 …]

This command is used to write a 4-byte value to a block in a card compatible with MIFARE Standard. User should perform succeed authentication to get the access right of the block before sending this command.

Command

Command	Class	INS	P1	P2	Lc	Data In
Write Value Block	FFh	D7h	00h	Block Number	05h	See below

Command Data

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
00h	4 Bytes Value with MSB first			

Example 1: Decimal –4 = {FFh, FFh, FFh, FCh}

VB_Value			
MSB			LSB
FFh	FFh	FFh	FCh

Example 2: Decimal 1 = {00h, 00h, 00h, 01h}

VB_Value			
MSB			LSB
00h	00h	00h	01h

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

5.3.4.2. Read Value Block [FF B1 …]

This command is used to read a 4-byte value from a valid value block in a card compatible with MIFARE Standard. User should perform succeed authentication to get the access right of the block before sending this command.

Command

Command	Class	INS	P1	P2	Le
Read Value Block	FFh	B1h	00h	Block Number	04h

Example 1: Decimal –4 = {FFh, FFh, FFh, FCh}

Value			
MSB			LSB
FFh	FFh	FFh	FCh

Example 2: Decimal 1 = {00h, 00h, 00h, 01h}

Value			
MSB			LSB
00h	00h	00h	01h

Response

Rsp Data	SW1 SW2	Meaning
4 Bytes Value with MSB first	90 00h	Succeed with data.
--	6X XXh	Fail.

5.3.4.3. Decrement/Increment Value [FF D7 …]

This command is used to decrement/Increment a 4-byte value from source block and stores the result to target block in a card compatible with MIFARE Standard. If user wants to store the result to the block same as source block, user can set the target block number equal to 0 or source block number. User should perform succeed authentication to get the access right of both source and target block before sending this command.

Command

Command	Class	INS	P1	P2	Lc	Data In
Decrement/Increment Value	FFh	D7h	Target Block#	Source Block#	05h	See below

Command Data

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
01h	4 Bytes Increment Value with MSB first			
02h	4 Bytes Decrement Value with MSB first			

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

5.3.4.4. Copy Value Block [FF D7 …]

This command is used to copy the value from source block to target block in a card compatible with MIFARE Standard. User should perform succeed authentication to get the access right of both source and target block before sending this command.

Command

Command	Class	INS	P1	P2	Lc	Data In
Copy Value Block	FFh	D7h	00	Source Block#	02h	See below

Command Data

Byte 0	Byte 1
03h	Target Block#

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	6X XXh	Fail.

5.3.5. Accessing PCSC-Compliant tags (ISO14443-4)

All ISO 14443-4 compliant cards (PICCs) understand the ISO 7816-4 APDUs. The ACR1552U reader just has to communicate with the ISO 14443-4 compliant cards by exchanging ISO 7816-4 APDUs and responses. The ACR1552U will handle the ISO 14443 Parts 1-4 Protocols internally.

MIFARE Classic (1K/4K), MIFARE Mini and MIFARE Ultralight tags are supported through the T=CL emulation. Just simply treat the MIFARE tags as standard ISO 14443-4 tags. For more information, please refer to **PCSC Pseudo APDU (with Proprietary Extension) for PICC**.

ISO 7816-4 APDU Format

Command	Class	INS	P1	P2	Lc	Data In	Le
ISO 7816 Part 4 Command					Length of the Data In		Expected length of the Response Data

ISO 7816-4 Response Format (Data + 2 bytes)

Response	Data Out		
Result	Response Data	SW1	SW2

Common ISO 7816-4 Response Codes

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	63 00h	The operation failed.

Typical sequence may be:

1. Present the tag and connect the PICC Interface.
2. Read/Update the memory of the tag.

To do this:

1. Connect the tag.

   The ATR of the tag is 3B 88 80 01 00 00 00 00 33 81 81 00 3Ah.

   In which,

   The Application Data of ATQB = 00 00 00 00, protocol information of ATQB = 33 81 81. It is an ISO 14443-4 Type B tag.

2. Send an APDU, Get Challenge.

   << 00 84 00 00 08h

   >> 1A F7 F3 1B CD 2B A9 58h [90 00h]

   Note: *For ISO 14443-4 Type A tags, the ATS can be obtained by using the APDU "FF CA 01 00 00h."*

Example:

// Read 8 bytes from an ISO 14443-4 Type B PICC (ST19XR08E)

APDU = {80 B2 80 00 08h}

Class	= 80h
INS	= B2h
P1	= 80h
P2	= 00h
Lc	= None
Data In	= None
Le	= 08h

Answer: 00 01 02 03 04 05 06 07h [$9000h]

5.3.6. Accessing FeliCa tags

For FeliCa access, the command is different from the one used in PCSC-compliant and MIFARE tags. The command follows the FeliCa specification with an added header.

FeliCa Command Format

Command	Class	INS	P1	P2	Lc	Data In
FeliCa Command	FFh	00h	00h	00h	Length of the Data In	FeliCa Command (start with Length Byte)

FeliCa Response Format (Data + 2 bytes)

Response	Data Out
Result	Response Data

Read Memory Block Example:

1. Connect the FeliCa.

   The ATR = 3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42h

   In which, 11 00 3Bh = FeliCa

2. Read FeliCa IDM.

   CMD = FF CA 00 00 00h

   RES = [IDM (8bytes)] 90 00h

   e.g., FeliCa IDM = 01 01 06 01 CB 09 57 03h

3. FeliCa command access.

   Example: “Read” Memory Block.

   CMD = FF 00 00 00 10 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   where:

   Felica Command = 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   IDM = 01 01 06 01 CB 09 57 03h

   RES = Memory Block Data

5.3.8. Accessing ISO15693 tags

This section shows the option commands for ISO15693, the firmware requirement are illustrate as follow:

- ACR1552U-M FW 1.03.00 or above
- ACM1552U-Y FW 2.03.00 or above
- ACM1552U-Z FW 2.03.00 or above

5.3.8.1. Read Single Block

This command retrieves one data block from the ISO15693 tag.

Command:

Command	Class	INS	P1	P2	LC	Data		Le
Read Single Block	FFh	FBh	00h	00h	02h	20h	Block Number	--/00h

Where:

Block Number 1 byte.

The data block number.

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

//Read NXP ICODE SLI card block 10 data

Command: = { FF FB 00 00 02 20 10 }

Response: = { XX XX XX XX 90 00 }

5.3.8.2. Write Single Block

This command write one data block to the ISO15693 tag.

Command:

Command	Class	INS	P1	P2	LC	Data			Le
Write Single Block	FFh	FBh	00h	00h	N+2h	21h	Block Number	Block Data	--/00h

Where:

Block Number 1 byte.

The data block number.

Block Data N bytes.

The data write to the block

LC 1 byte.

Base on the length of block + 2

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

//Write NXP ICODE SLI card block 10 data

Command: = { FF FB 00 00 06 21 10 11 12 13 14}

Response: = { 90 00 }

5.3.8.3. Read Multiple Blocks

This command retrieves data blocks from the ISO15693 tag.

Command:

Command	Class	INS	P1	P2	LC	Data			Le
Read Multiple Blocks	FFh	FBh	00h	00h	03h	23h	First Block Number	Number of Blocks	--/00h

Where:

First Block Number 1 byte.

The starting data block number.

Number of Blocks 1 byte.

The number of blocks in the request is one less than the number of block security status that the tags will return in its response.

Number of Blocks = The number of blocks in the request - 1

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

//Get multiple blocks security status from 0x10 to 0x12. 0x03 consecutive blocks of NXP ICODE SLI card.

Command: = { FF FB 00 00 03 23 10 02 }

Response: = { XX XX XX XX XX XX XX XX XX XX XX XX 90 00 }

5.3.8.4. Write Multiple Blocks

This command write data blocks to the ISO15693 tag.

Command:

Command	Class	INS	P1	P2	LC	Data				Le
Write Multiple Blocks	FFh	FBh	00h	00h	N+3h	24h	First Block Number	Number of Blocks	Block Data	--/00h

Where:

First Block Number 1 byte.

The starting data block number.

Number of Blocks 1 byte.

The number of blocks in the request is one less than the number of block security status that the tags will return in its response.

Number of Blocks = The number of blocks in the request – 1

Block Data N bytes.

The data write to the blocks

LC 1 byte.

Base on the length of block data + 3

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

5.3.8.5. Lock Block

The Lock block command will lock permanently the requested block and report the success of the operation in the response

Command:

Command	Class	INS	P1	P2	LC	Data		Le
Lock Blocks	FFh	FBh	00h	00h	02h	22h	Block Number	--/00h

Where:

Block Number	1 byte. The data block number.

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

Command: = { FF FB 00 00 02 22 10 }

Response: = { 90 00 }

5.3.8.6. Get System Information

The Get System Information command will send back the system information form the tag.

Command:

Command	Class	INS	P1	P2	LC	Data	Le
Get System Information	FFh	FBh	00h	00h	01h	2Bh	--/00h

Get System Information Response Format

Response	Data Out							
Result	Info Flags	UID	DSFID	AFI	Memory Size	IC Reference	SW1	SW 2

Where:

Info Flags - 1 Byte

Bit	Value	Description
Bit 0	0	DSFID not present
	1	DSFID present
Bit 1	0	AFI not present
	1	AFI present
Bit 2	0	Memory size not present
	1	Memory size present
Bit 3	0	IC reference not present
	1	IC reference present
Bit 4 ~7	0	RFU

UID - 8 Byte

DSFID - 1 Byte

AFI - 1 Byte

Memory Size - 2 Byte

Byte	Description
0	Number of blocks - 1 (The actual Number of blocks = Number of blocks + 1
1	Block size in Bytes - 1 (The actual block size = Block size in Bytes +1)

IC Reference - 1 Byte

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

Command: = { FF FB 00 00 01 2B }

Response: = { XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00}

5.3.8.7. Get Multiple Blocks Security Status

The Get Multiple blocks Security Status will send back the block security status

Command:

Command	Class	INS	P1	P2	LC	Data			Le
Get Multiple Blocks Security Status	FFh	FBh	00h	00h	03h	2Ch	First Block Number	Number of Blocks	--/00h

Where:

First Block Number 1 byte.

The starting data block number.

Number of Blocks 1 byte.

The number of data block security status will be read. The number of blocks in the request is one less than the number of block security status that the tags will return in its response.

Number of Blocks = The number of blocks in the request - 1

Get System Information Response Format

Response	Data Out		
Result	Block Security Status	SW1	SW2

Where:

Block Security Status Each block for 1 byte.

00h: Unlocked

01h: Locked

Response Code

Results	SW1 SW2	Meaning
Success	90 00h	The operation was completed successfully.
Error	64 XXh	Fail. XX is the error code from the tag

Examples:

//Get multiple blocks security status from 0x10 to 0x12. 0x03 consecutive blocks.

Command: = { FF FB 00 00 03 2C 10 02 }

Response: = { XX XX XX 90 00 }

5.3.9. Supported PICC ATR

The following PICC type/technology are supported by default. The following ATR is returned to CCID Host on PC_to_RDR_IccPowerOn Command if the card is presented to the reader.

Card Type/Technology	ATR
MIFARE Std 1k2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
MIFARE Std 4k2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 02 00 00 00 00 69
MIFARE UltraLight2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 03 00 00 00 00 68
MIFARE Plus SL1 2k2	Default: Same as MIFARE Std 1k Alternated: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 36 00 00 00 00 5D
MIFARE Plus SL1 4k2	Default: Same as MIFARE Std 4k Alternated: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 37 00 00 00 00 5C
MIFARE Plus SL2 2k	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 38 00 00 00 00 53
MIFARE Plus SL2 4k	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 39 00 00 00 00 52
MIFARE UltraLight C2	Default: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3A 00 00 00 00 51 Alternated: Same as MIFARE UltraLight
SmartMX with MIFARE Std 1k Emulation2	Default: Same as MIFARE Std 1k Alternated: Same as ISO14443-4, Type A
SmartMX with MIFARE Std 4k Emulation[2]	Default: Same as MIFARE Std 4k Alternated: Same as ISO14443-4, Type A

ISO14443-4, Type A	3B 8n 80 01 T1 .. Tn Tck
	
	n = Number of Historical bytes in ATS
	T1 .. Tn = Historical bytes in ATS
	Tck = XOR of 8n 80 01 T1 .. Tn

ISO14443-4, Type B	3B 88 80 01 T1 .. T8 Tck T1 .. T4 = Application Data in ATQB T5 .. T7 = Protocol Info in ATQB T8 = MBLI in ATA Tck = XOR of 88 80 01 T1 .. T8
FeliCa	3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42
ISO15693-3 Generic	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 00 00 00 00 00 63
Infineon My-D Vicinity (SRF55Vxxx)	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 0E 00 00 00 00 6D
ST LRI	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 13 00 00 00 00 70
NXP I-Code SLI	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 14 00 00 00 00 77
NXP I-Code SLIX/SLIX2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 35 00 00 00 00 56

[2] Refer to "Param 2" in Set Operation Mode Escape command for configuration and drawback of the alternated ATR definition.

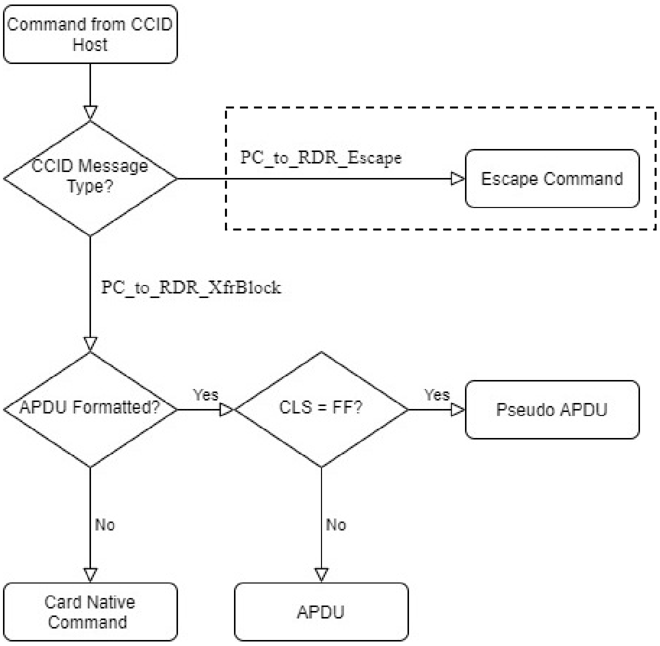
Card Type/Technology	ATR
PicoPass 2K	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 17 00 00 00 00 79
PicoPass 2KS	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 18 00 00 00 00 76
PicoPass 16K	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 19 00 00 00 00 77
PicoPass 16KS	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1A 00 00 00 00 74
PicoPass 16K (8 x 2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1B 00 00 00 00 75
PicoPass 16KS (8 x 2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1C 00 00 00 00 72
PicoPass 32KS (16 + 16)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1D 00 00 00 00 73
PicoPass 32KS (16 + 8x2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1E 00 00 00 00 70
PicoPass 32KS (8x2 + 16)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1F 00 00 00 00 71
PicoPass 32KS (8x2 + 8x2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 20 00 00 00 00 4E

In order to reduce response time for generic application, the support of following PICC type/technology are disabled by default. User could enable the support of each Type/Technology by “Set operation Mode” Escape command. The following ATR is returned to CCID Host on PC_to_RDR_IccPowerOn Command if the card is presented to the reader and the corresponding Type/Technology is enabled.

Card Type/Technology	ATR
SRI (SRIX4K/SRT512)	3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 07 00 00 00 00 69
Topaz	3B 8F 80 01 80 4F 0C A0 00 00 03 06 02 00 30 00 00 00 00 5A
PicoPass 2K	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 17 00 00 00 00 75
PicoPass 2KS	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 18 00 00 00 00 7A
PicoPass 16K	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 19 00 00 00 00 7B
PicoPass 16KS	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1A 00 00 00 00 78
PicoPass 16K (8 x 2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1B 00 00 00 00 79
PicoPass 16KS (8 x 2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1C 00 00 00 00 7E
PicoPass 32KS (16 + 16)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1D 00 00 00 00 7F
PicoPass 32KS (16 + 8x2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1E 00 00 00 00 7C
PicoPass 32KS (8x2 + 16)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1F 00 00 00 00 7D
PicoPass 32KS (8x2 + 8x2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 20 00 00 00 00 42
Innovatron	3B 88 80 01 80 4F 05 F0 49 4E 4E 4F 35
CTS	3B 87 80 01 80 4F 04 F0 43 54 53 79

6.0. Escape Command

Escape Command is send by PC_to_RDR_Escape (corresponding to SCardControl() with SCARD_CTL_CODE(3500) in PCSC API. . After the command processing, the Reader will send back the response by RDR_to_PC_Escape Message.



The following commands are provided to configure PCD/NFC and to access special function of the reader. CCID Host could send these commands to reader by using CCID Message PC_to_RDR_Escape (corresponding to SCardControl() with SCARD_CTL_CODE(3500) in PCSC API). After receiving of an Escape Command, it will be interpreted to perform various operations and then generate a response to send back to CCID Host.

Note:
Should send these commands under correct interface. For example, E0 00 00 25 01 00 (Section 6.4.1.1) should send through PICC interface (Section 6.4.1).

6.1. Escape Command for PICC

6.1.1. RF Control [E0 00 00 25 01 …]

This command is used to set the RF control.

Command

Command	Class	INS	P1	P2	Lc	Data Out
RF Control	E0h	00h	00h	25h	01h	RF status

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	RF status

RF Status: 1 Byte

RF status	Description
00h	RF Off
01h	RF On, with Polling
02h	RF On, without Polling

Default Setting – 01h (RF On, with Polling)

6.1.2. Get PCD/PICC Status [E0 00 00 25 00]

This command is used to get the PCD/PICC status

Command

Command	Class	INS	P1	P2	Le
Get PCD/PICC Status	E0h	00h	00h	25h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Get PCD/PICC Status

PCD/PICC Status: 1 Byte

RF status	Description
00h	RF Off
01h	No PICC
02h	PICC Ready
03h	PICC Selected/Activated
FFh	Error

6.1.3. Get Polling/ATR Option [E0 00 00 23 00]

This command is used to set/get the Polling Option but save the setting without another command. This command should only be used for initial reader configuration.

Command

Command	Class	INS	P1	P2	Le
Get Polling/ATR Option	E0h	00h	00h	23h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	PICC Polling/ATR Option

6.1.4. Set Polling/ATR Option [E0 00 00 23 01 …]

This command is used to set the polling option.

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set Polling/ATR Option	E0h	00h	00h	23h	01h	PICC Polling/ATR Option

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	PICC Polling/ATR Option

PICC Polling/ATR Option - 1 Byte

Operating Parameter	Parameter	Description	Option
Bit 0	Enable Polling	The Tag Types to be detected during PICC Polling.	1 = Detect 0 = Skip
Bit 1	Enable RF Off		
Bit 2	RFU		
Bit 3	Enable extra MIFARE type identification for Part 3 card in ATR	The Tag Types to be detected during PICC Polling.	1 = Detect 0 = Skip
Bit 4 ~ 5	RF Off Interval		See below
Bit 6	RFU		
Bit 7	Enable Part 4 ATR for SmartMX/JCOS card with MIFARE emulation	The Tag Types to be detected during PICC Polling.	1 = Detect 0 = Skip

RF Off Interval – 2 Bit **Case 1:** Disabled RF Off (Bit 1 = 0)

Operating Parameter		USB Active (D0)	USB Suspend (D2)
Bit 5	Bit 4		
0	0	No RF Off	250 ms
0	1		500 ms
1	0		1000 ms
1	1		2500 ms

Case 2: Enabled RF Off (Bit 1 = 1)

Operating Parameter		USB Active (D0)	USB Suspend (D2)
Bit 5	Bit 4		
0	0	250 ms	500 ms
0	1	500 ms	1000 ms
1	0	1000 ms	2500 ms
1	1	2500 ms	2500 ms

Default Setting – 8Bh (Enabled Polling, Enabled RF Off, Enabled extra MIFARE type identification for Part 3 card in ATR, RF Off Interval[00], Enabled Part 4 ATR for SmartMX/JCOS card with MIFARE emulation)

6.1.5. Get PICC Polling Type [E0 00 01 20 00]

This command is used to get the allowed Technology/Polling Type but save the setting without another command. This command should only be used for initial reader configuration.

Command

Command	Class	INS	P1	P2	Le
Get PICC Polling Type	E0h	00h	01h	20h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In	
						Byte 1	Byte 0
Result	E1h	00h	00h	00h	02h	PICC Polling Type	

6.1.6. Set PICC Polling Type [E0 00 01 20 02 …]

This command is used to set the PICC polling type.

Command

Command	Class	INS	P1	P2	Lc	Data Out	
						Byte 1	Byte 0
Set PICC Polling Type	E0h	00h	01h	20h	02h	PICC Polling Type	

Response Code

Response	Class	INS	P1	P2	Le	Data In	
						Byte 1	Byte 0
Result	E1h	00h	00h	00h	02h	PICC Polling Type	

PICC Polling Type - 2 Byte, Bit Mask of following

Bytes	Operating Parameter	Parameter	Description	Option
Byte 1	Bit 0	ISO 14443A Type A	The Tag Types to be detected during PICC Polling. RFU bit should be set to 0.	1 = Detect 0 = Skip
	Bit 1	ISO 14443A Type B		
	Bit 2	FeliCa		
	Bit 3	RFU		
	Bit 4	Topaz		
	Bit 5	Innovatron		
	Bit 6	SRI/SRIX		
	Bit 7	RFU		
Byte 0	Bit 0	Picopass (ISO14443B)		

Bytes	Operating Parameter	Parameter	Description	Option
	Bit 1	Picopass (ISO15693)		
	Bit 2	ISO15693		
	Bit 3	CTS		
	Bit 4-7	RFU		

Default Setting – Byte 1: 07h (ISO14443 Type A, ISO14443 Type B, FeliCa)

Byte 0: 05h (Picopass (ISO14443B), ISO15693)

Example:

Command: E0 00 01 20 02 07 05

Response: E1 00 00 00 02 07 05

Polling Type: Byte 1 = 07h = 0000 0111b = ISO14443 Type A, ISO14443 Type B, FeliCa

Byte 0 = 05h = 0000 0101b = Picopass (ISO14443B), ISO15693

6.1.7. Get Auto PPS [E0 00 00 24 00]

Whenever a PICC is recognized, the reader will try to change the communication speed between the PCD and PICC as defined by the maximum connection speed. If the card does not support the proposed connection speed, the reader will try to connect the card with a slower speed setting.

Command

Command	Class	INS	P1	P2	Le
Get Auto PPS	E0h	00h	00h	24h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	Max Speed	Current Speed

6.1.8. Set Auto PPS [E0 00 00 24 01 …]

This command is used to set the auto PPS.

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set Auto PPS	E0h	00h	00h	24h	01h	Max Speed

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	Max Speed	Current Speed

Speed of PPS

Speed	Description
00h	106 kbps; equal to No Auto PPS
01h	212 kbps
02h	424 kbps
03h	848 kbps

Default Setting – 02h (424 kbps)

Notes:

1. Normally, the application should know the maximum connection speed of the PICCs being used. The environment also affects the maximum achievable speed. The reader just uses the proposed communication speed to talk with the PICC. The PICC will become inaccessible if the PICC or environment does not meet the requirement of the proposed communication speed.

2. If the higher speed setting affects the performance of the reader, please switch back to a lower speed setting.

6.1.9. Read PICC Type [E0 00 00 35 00]

This command is used to read the PICC type.

command

Command	Class	INS	P1	P2	Le
Get PICC Type	E0h	00h	00h	35h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	Type	Status

Type: 1 Byte

Type	Description
CCh	No PICC
04h	Topaz
10h	MIFARE
11h	FeliCa
20h	Type A, Part 4
23h	Type B, Part 4
25h	Innovatron
28h	SRIX
30h	PicoPass
FFh	Other

Status: 1 Byte

Status	Description
00h	RF Off

Status	Description
01h	No PICC
02h	PICC Ready
03h	PICC Selected/Activated
FFh	Error

6.1.10. Get RF Power Setting [E0 00 00 50 00]

This command is used to read the RF Power Setting, the firmware requirement are illustrate as follow:

- ACR1552U-M FW 1.03.03 or above
- ACM1552U-Y FW 2.03.03 or above
- ACM1552U-Z FW 2.03.03 or above

Command

Command	Class	INS	P1	P2	Le
Get RF Power Setting	E0h	00h	00h	50h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E0h	00h	00h	00h	01h	RF Power

6.1.11. Set RF Power Setting [E0 00 00 50 01 …]

This command is used to set the PICC polling type, the firmware requirement is the same as Get RF Power Setting.

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set RF Power Setting	E0h	00h	01h	50	01h	RF Power

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E0h	00h	00h	00h	01h	RF Power

Percentage mode

RF Power - 1 Byte

Parameter	Description
00h	Disable manual RF Power setting
01h	20%
02h	40%
03h	60%
04h	80%
05h	100%

Default Setting – 00h

* RF Power value in Percentage mode may not have effective due to hardware limitation.

6.1.12. Escape Command for PICC – HID Keyboard

6.1.12.1. Get Output Format [E0 00 00 90 00]

This command is used to get output format.

Command

Command	Class	INS	P1	P2	Le
Get Output Format	E0h	00h	00h	90h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	Output Format	Output Order

6.1.12.2. Set Output Format [E0 00 00 90 02 …]

This command is used to set output format.

Command

Command	Class	INS	P1	P2	Lc	Data Out	
Set Output Format	E0h	00h	00h	90h	02h	Output Format	Output Order

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	Output Format	Output Order

Output Format: 1 Byte

Operating Parameter	Parameter	Description	Option
Bit 7 ~ 4	Letter Case	The Tag Types to be detected during PICC Polling.	1 = Detect 0 = Skip
Bit 3 ~ 0	Display Mode		

Output Order: 1 Byte

Status	Description
00h	Default order (UID Byte 0, UID Byte 1 … UID Byte N) Example: aa cc bb dd (original /actual UID order)
01h	Reverse order (UID Byte N, UID Byte N-1 … UID Byte 0) Example: dd bb cc aa (reverse the UID order)

Letter Case: Upper 4 Bits (Bit 7 ~ 4)

Status (From bit 7~4)	Description (Don't care about x bit)
1xxx	Reserved
00x0	Lowercase
00x1	Uppercase
000x	Only Support 4 bytes UID
001x	Support 4, 7, 8, 10 bytes UID

Display Mode: Lower 4 Bits (Bit 3 ~ 0)

Status (From bit 7~4)	Description (Don't care about x bit)
0h	Hex
1h	Dec (byte by byte)
2h	Dec
3h	6H-6H
4h	8H-8H
5h	10H-10H
6h	14H-14H
7h	20H-20H
8h	6H-8D
9h	6H-10D
Ah	8H-10D
Bh	10H-14D
Ch	2H4H-8D
Dh	14H-17D

6.1.12.3. Get Character at Start, Between, at End UID [E0 00 00 91 00]

This command is used to get character at Start, Between, End UID.

Command

Command	Class	INS	P1	P2	Le
Get Character of UID	E0h	00h	00h	91h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In		
Result	E1h	00h	00h	00h	03h	Between	End	Start

6.1.12.4. Set Character at Start, Between, at End UID [E0 00 00 91 03 …]

This command is used to set character at Start, Between, End UID.

Command

Command	Class	INS	P1	P2	Lc	Data Out		
Set Character of UID	E0h	00h	00h	91h	03h	Between	End	Start

Response Code

Response	Class	INS	P1	P2	Le	Data In		
Result	E1h	00h	00h	00h	03h	Between	End	Start

Between: 1 Byte (The character between each UID)

Status	Description
FFh	No character in between
Other	Refer to Universal Serial Bus (USB) HID Usage Tables

End: 1 Byte (The character at the end of output)

Status	Description
FFh	No character in between
Other	Refer to Universal Serial Bus (USB) HID Usage Tables

Start: 1 Byte (The character at the start of output)

Status	Description
FFh	No character in between
Other	Refer to Universal Serial Bus (USB) HID Usage Tables

Notes:

1. *only the characters “;” “,” “,” “,” “-“ are supported in the AZERTY keyboard layout for the characters in between. Zero (0) and Backspace are NOT supported.*

6.1.12.5. Get Keyboard Layout Language [E0 00 00 92 00]

This command is used to get keyboard layout language.

Command

Command	Class	INS	P1	P2	Le
Get Keyboard Layout Language	E0h	00h	00h	92h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Keyboard Layout Language

6.1.12.6. Set Keyboard Layout Language [E0 00 00 92 01 …]

This command is used to set keyboard layout language.

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set Keyboard Layout Language	E0h	00h	00h	92h	01h	Keyboard Layout Language

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Keyboard Layout Language

Keyboard Layout Language: 1 Byte

Status	Description
00h	English
01h	French
02h	Reserved
03h	Lithuanian

6.1.12.7. Get Host Interface [E0 00 00 93 00]

This command is used to get host interface

Command

Command	Class	INS	P1	P2	Le
Get Host Interface	E0h	00h	00h	93h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Host Interface

6.1.12.8. Set Host Interface [E0 00 00 93 01 …]

This command is used to set host interface command

Command	Class	INS	P1	P2	Lc	Data Out
Set Host Interface	E0h	00h	00h	93h	01h	Host Interface

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Host Interface

Host Interface: 1 Byte

Status	Description
00h	Only HID Keyboard
01h	Only CCID Reader
02h	HID Keyboard + CCID Reader

6.1.13. Escape Command for PICC – Card Emulation

6.1.13.1. Enter Card Emulation Mode [E0 00 00 40 03 …]

This command is used to set the reader into card emulation mode in order to emulate a MIFARE Ultralight or a FeliCa Card.

Note: *Lock byte is not supported in emulated MIFARE Ultralight. UID is user programmable.*

Command

Command	Class	INS	P1	P2	Lc	Data Out		
Enter Card Emulation Mode	E0h	00h	00h	40h	03h	NFC Mode	00h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	03h	NFC Mode

NFC Device Mode: 3 Byte

Status	Description
02h	NFC Forum Type 2 Tag Mode
03h	FeliCa
Other	Card Read/Write Mode

Note: *Please enter to Card Read/Write mode before switching to different card emulation mode. The response will be showed after the Card Emulation Mode initial is done.*

Byte Number	0	1	2	3	Byte Address access by USB
Serial Number	SN0	SN1	SN2	SN3	Nil
Reserved	Reserved	Reserved	Reserved	Reserved	Nil
Internal/Lock	Reserved	Internal	Lock0	Lock1	Nil
Data read/write	Data0	Data1	Data2	Data3	0-3
Data read/write	Data4	Data5	Data6	Data7	4-7
Data read/write	Data8	Data9	Data10	Data11	8-11
Data read/write	Data12	Data13	Data14	Data15	12-15
Data read/write	Data16	Data17	Data18	Data19	16-19
Data read/write	Data20	Data21	Data22	Data23	20-23
Data read/write	Data24	Data25	Data26	Data27	24-27
Data read/write	Data28	Data29	Data30	Data31	28-31
Data read/write	Data32	Data33	Data34	Data35	32-35
Data read/write	Data36	Data37	Data38	Data39	36-39
Data read/write	Data40	Data41	Data42	Data43	40-43
Data read/write	Data44	Data45	Data46	Data47	44-47
Data read/write	Data48	Data49	Data50	Data51	48-51
Data read/write	Data52	Data53	Data54	Data55	52-55
Data read/write		…			…
Data read/write	Data1984	Data1985	Data1986	Data1987	1984~1987

Accessible area (1988 bytes)

Table 6: NFC Forum Type 2 Tag Memory Map (2000 bytes)

Memory	1 Block data (16 Byte)	Byte Address access by USB
Data read/write	Block 0	0-15
Data read/write	Block 1	16-31
Data read/write	Block 2	32-47
Data read/write	Block 3	48-63
Data read/write	Block 4	64-79
Data read/write	Block 5	80-95
Data read/write	Block 6	96-111
Data read/write	Block 7	112-127
Data read/write	Block 8	128-143
Data read/write	Block 9	144-159

Table 7: FeliCa Memory Map (160 bytes)

Where:

Default: Block 0 data: {10h, 01h, 01h, 00h, 09h, 00h, 00h, 00h, 00h, 00h, 01h, 00h, 00h, 00h, 00h, 1Ch}

Default Block 0 data NFC Type3 Tag Attribute Information Block

Notes:

1. *FeliCa card emulation support Read/Write without Encryption*
2. *FeliCa Card Identification Number in IDm is user programmable while Manufacturer Code is fixed at (03 88).*

6.1.13.2. Read Card Emulation Data (NFC Forum Type 2 Tag) [E0 00 00 60 04 …]

This command is used to read the emulated card content.

Command

Command	Class	INS	P1	P2	Lc	Data In			
Read Card Emulation Data	E0h	00h	00h	60h	04h	00h	NFC Mode	Start Offset	Length

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	Length	Data

Start Offset: 1 Byte – Address start from Data0 in **Table 6**

Length: 1 Byte – No. of byte

6.1.13.3. Write Card Emulation Data (NFC Forum Type 2 Tag) [E0 00 00 60 …]

This command is used to write the emulated card content.

Command

Command	Class	INS	P1	P2	Lc	Data In				
Write Card Emulation Data	E0h	00h	00h	60h	Length + 04h	01h	NFC Mode	Start Offset	Length	Data

Response Code

Response	Class	INS	P1	P2	Le	Data In		
Result	E1h	00h	00h	00h	03h	Length	90h	00h

NFC Device Mode: 1 Byte

Status	Description
02h	NFC Forum Type 2 Tag Mode
03h	FeliCa
Other	Card Read/Write Mode

Start Offset: 1 Byte – Address start from Data0 in **Table 6**

Length: 1 Byte – No. of byte

6.1.13.4. Read Card Emulation Data (NFC Forum Type 2 Tag)) (Extended)

This command is used to read the emulated card content.

Command

Command	Class	INS	P1	P2	Lc		Data In			
Read Card Emulation Data	E0h	00h	01h	60h	05h	00h	NFC Mode	Start Offset Bit[15:8]	Start Offset Bit[7:0]	Length

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	Length	Data

Start Offset: 2 Byte – Address start to read from SN0 in **Table 6**

Length: 1 Byte – No. of byte to read

6.1.13.5. Write Card Emulation Data (NFC Forum Type 2 Tag) (Extended)

This command is used to write the emulated card content.

Command

Command	Class	INS	P1	P2	Lc		Data In				
Write Card Emulation Data	E0h	00h	01h	60h	Length + 05h	01h	NFC Mode	Start Offset Bit[15:8]	Start Offset Bit[7:0]	Length	Data

Response Code

Response	Class	INS	P1	P2	Le	Data In		
Result	E1h	00h	00h	00h	03h	Length	90h	00h

NFC Device Mode: 1 Byte

Status	Description
02h	NFC Forum Type 2 Tag Mode
Other	Card Read/Write Mode

Start Offset: 2 Byte – Address start to write from SN0 in **Table 6**

Length: 1 Byte – No. of byte to write

6.1.13.6. Set Card Emulation of NFC Forum Type 2 Tag ID [E0 00 00 61 03 …]

This command sets the UID of the emulated MIFARE Ultralight card.

Command

Command	Class	INS	P1	P2	Lc	Data In
Set Card Emulation Lock Data	E0h	00h	00h	61h	03h	3 bytes UID

Response Code

Response	Class	INS	P1	P2	Le	Data In	
Result	E1h	00h	00h	00h	02h	90h	00h

6.1.13.7. Set Card Emulation Lock Data in NFC [E0 00 00 65 01 …]

This command sets the lock for card emulation data in NFC communication. If the data is locked, it is protected from being overwritten via NFC.

Command

Command	Class	INS	P1	P2	Lc	Data In
Set Card Emulation Lock Data	E0h	00h	00h	65h	01h	Lock

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Lock

Lock: 1 Byte – Protect the data from being overwritten via NFC

Operating Parameter	Parameter	Description	Option
Bit 7 ~ 2	Reserved	Reserved	
Bit 1	FeliCa Lock Enable	Data cannot be modified via NFC. The data can still be modified by using the USB escape command.	0: Lock disable 1: Lock enable
Bit 0	NFC Forum Type 2 Tag Enable		

6.1.13.8. Set Card Emulation FeliCa IDm [E0 00 00 64 06 …]

This command sets the 6-byte FeliCa Card Identification number on emulated FeliCa card.

Command

Command	Class	INS	P1	P2	Lc	Data In
Set Card Emulation FeliCa IDm	E0h	00h	00h	64h	06h	IDm

Response Code

Response	Class	INS	P1	P2	Le	Data Out
Result	E1h	00h	00h	00h	06h	IDm

Where:

IDm	6 bytes.

6.1.13.9. Get Card Emulation Status [E0 00 00 69 00]

This command is used to get the status of card emulation data in NFC communication.

Command

Command	Class	INS	P1	P2	Lc
Get Card Emulation Status	E0h	00h	00h	69h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Status

Status: 1 Byte

Operating Parameter	Mode	Description
Bit 7 ~ 6	Reserved	Reserved
Bit 5	EmulatedCard is activated	1 = Activated
Bit 4	EmulatedCard is removed	1 = Card is removed
Bit 3	EmulatedCard is read all	1 = All data is read
Bit 2	EmulatedCard is read	1 = Data is read
Bit 1	EmulatedCard is written	1 = Data is written
Bit 0	EmulatedCard is detected	1 = Card is detecting

6.1.13.10. Example Command Set of Emulating NFC Forum Type 2 Tag Mode

The command set is to trigger ACS website https://www.acs.com.hk by using ACR1552U to emulate as the NFC forum type 2 tag mode. The steps are showed below:

1. Enter the card emulation mode with below command:
   - Send Enter Card Emulation Mode

     E0 00 00 40 03 02 00 00
2. Write the NDEF data with below command:
   - Send Write Card Emulation Data (NFC Forum Type 2 Tag)

     E0 00 00 60 1C 01 02 00 18 E1 10 F4 00 03 0F D1 01 0B 55 02 61 63 73 2E 63 6F 6D 2E 68 6B FE 00 00

Notes:

For more detailed information and specifications related to the NDEF (NFC Data Exchange Format), I would recommend referring to the NDEF specification. It provides comprehensive guidelines and details about the structure and usage of NDEF records, which are commonly used in NFC data exchange. The NDEF specification will provide a deeper understanding of how to interpret and utilize the NDEF command and data in the context of the ACR1552U device.

6.1.14. Escape Command for PICC – Discovery Mode

6.1.14.1. Enter Discovery Mode [E0 00 00 6A 01 …]

This command is used to enter the discovery mode.

Command

Command	Class	INS	P1	P2	Lc	Data Out
Enter Discovery Mode	E0h	00h	00h	6Ah	01h	Discovery Mode

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	Discovery Mode

Discovery Mode: 1 Byte

Status	Description
00h	Card Reader Mode
02h	NFC Forum Type 2 Tag Mode
03h	FeliCa

6.2. Escape Command for Peripheral Control and Other

6.2.1. Get Firmware Version [E0 00 00 18 …]

This command is used to get reader's firmware message.

Command

Command	Class	INS	P1	P2	Le
Get Firmware Version	E0h	00h	00h	18h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	Length of Firmware Version	Firmware Version

Example:
Command: E0 00 00 18 00
Response Code: E1 00 00 00 14 41 43 52 31 35 35 32 20 52 20 46 57 20 31 2E 30 30 2E 30 30
Firmware Version in Hex: 41 43 52 31 35 35 32 20 52 20 46 57 20 31 2E 30 30 2E 30 30
Firmware Version in ASCII: ACR1552 R FW 1.00.00

6.2.2. Get Serial Number [E0 00 00 33 00]

This command is used to get the serial number.

Command

Command	Class	INS	P1	P2	Le
Get Serial Number	E0h	00h	00h	33h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data out
Result	E1h	00h	00h	00h	Length of Serial No.	Serial No.

6.2.3. Set S/N in USB Descriptor [E0 00 00 F0]

This command is used to Set S/N in USB Descriptor.

Command

Command	Class	INS	P1	P2	Le	Data In	
Set S/N in USB Descriptor	E0h	00h	00h	F0h	02h	00h	Enable SN in USB Descriptor

Response Code

Response	Class	INS	P1	P2	Le	Data Out		
Result	E1h	00h	00h	00h	03h	Enable SN in USB Descriptor	90h	00h

Enable SN in USB Descriptor (1 byte)

Enable SN in USB Descriptor	Description
00h	Disable SN in USB Descriptor
01h	Enable SN in USB Descriptor

6.2.4. Set Buzzer Control - Single Time [E0 00 00 28 01 …]

This command is used to set a single buzzer

Command

Command	Class	INS	P1	P2	Lc	Data Out
Buzzer Control	E0h	00h	00h	28h	01h	BUZ Status

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	BUZ Status

Buzzer Status (1 byte)

Buzzer Status	Description
00h	Off
01 ~ FFh	On with duration in 10ms unit

6.2.5. Set Buzzer Control - Repeatable [E0 00 00 28 03 …]

This command is used to set period of buzzer

Command

Command	Class	INS	P1	P2	Lc	Data Out
Buzzer Control	E0h	00h	00h	28h	03h	BUZ Status

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	03h	BUZ Status

Buzzer Status (3 byte)

Operating Parameter	Buzzer Status	Description
Param 1 – Byte 0	On Time Period	01 ~ FF: On Duration in 10ms unit
Param 2 – Byte 1	Off Time Period	01 ~ FF: Off Duration in 10ms unit
Param 3 – Byte 2	Time for Repeating	01 ~ FF: Number to Repeat

6.2.6. Get LED Status [E0 00 00 29 00]

This command is used to get the current LED status

Command

Command	Class	INS	P1	P2	Le
Get LED Status	E0h	00h	00h	29h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	LED Status

6.2.7. Set LED Control [E0 00 00 29 01 …]

This command is used to set LED control

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set LED Control	E0h	00h	00h	29h	01h	LED Status

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	LED Status

LED Status (1 byte)

LED Status	Description
Bit 0: Blue LED	1 = On; 0 = Off
Bit 1: Green LED	1 = On; 0 = Off
Bit 2-7: RFU	Other

6.2.8. Get UI Behaviour [E0 00 00 21 00]

This command is used to get the PCD UI Behaviour but save the setting without another command. This command should only be used for initial reader configuration.

Command

Command	Class	INS	P1	P2	Le
Get PICC UI Behaviour	E0h	00h	00h	21h	00h

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	PICC UI Behaviour

6.2.9. Set UI Behaviour [E0 00 00 21 01 …]

This command is used to set the PICC UI behaviour.

For UI Behaviour bit 1 and bit 2 are available for the following firmware:

- ACR1552U-M FW 1.03.05 or above
- ACM1552U-Y FW 2.03.05 or above
- ACM1552U-Z FW 2.03.05 or above

Command

Command	Class	INS	P1	P2	Lc	Data Out
Set PICC UI Behaviour	E0h	00h	00h	21h	01h	PICCUI Behaviour

Response Code

Response	Class	INS	P1	P2	Le	Data In
Result	E1h	00h	00h	00h	01h	PICC UI Behaviour

UI Behaviour - 1 Byte, Bit Mask of following

Operating Parameter	Parameter	Description	Option
Bit 0	Accessing(LED Fast Blinking)	The UI behaviour of the reader	1 = Enable 0 = Disable
Bit 1	PICC Polling Status LED		
Bit 2	PICC Activation Status LED		
Bit 3	Presence Event (Short Buzzer Beep)		
Bit 4	Card Removal Event (Short Buzzer Beep)		

Default Setting For PICC – 0Fh

Notes:
1. The Get/Set UI behaviour are excluding on SAM interface.

Appendix A. NDEF Message

This section shows how to use NDEF message to encode the URL onto the NTag.

For the data format, please refer to NFC Forum NFC Data Exchange Format (NDEF) Specifications 1.0.

Example:

NDEF Message = {D1 02 0F 53 70 D1 01 0B 55 01 61 63 73 2E 63 6F 6D 2E 68 6Bh}

Offset	Content	Length	Description
0	D1	1	NDEF header. TNF = 01h, SR=1, MB=1, ME=1
1	02	1	Record name length (2 bytes)
2	0F	1	Length of the Smart Poster data (15 bytes)
3	53 70 (“Sp”)	2	Record name
5	D1	1	NDEF header. TNF = 01h, SR=1, MB=1, ME=1
6	01	1	Record name length (1 byte)
7	0B	1	The length of the URI payload (11 bytes)
8	55 (“U”)	1	Record type: “U”
9	01	1	Abbreviation: “http://www.”
10	61 63 73 2E 63 6F 6D 2E 68 6B	10	The URL itself. “acs.com.hk”